Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра інженерії
програмного забезпечення
в енергетиці (ІПЗЕ)

Системи керування версіями

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	Перший (бакалаврський)
Галузь знань	12 Інформаційні технології
Спеціальність	121 Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці
Статус дисципліни	Вибіркова
Форма навчання	Очна(денна)
Рік підготовки, семестр	4 курс осінній семестр
Обсяг дисципліни	4 кредитів ЄКТС/120 год (36 год. лекції, 18 год. практичні заняття, 66 год. самостійна робота)
Семестровий контроль/ контрольні заходи	Залік
Розклад занять	Згідно розкладу на весняний семестр поточного навчального року (http://roz.kpi.ua/)
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: старший викладач, Колумбет В.П., kvplinux@gmail.com, тел. 093-405-35-39 Практичні: старший викладач, Колумбет В.П., kvplinux@gmail.com, тел. 093-405-35-39
Розміщення курсу	Кампус

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

В цій дисципліні детально вивчається концепції, принципи та методи інформаційних технологій. Етапи розробки програмного забезпечення. Міжнародні стандарти програмної інженерії. Моделі та процеси, які впроваджуються в розробленні програмного забезпечення. Вимоги до інформаційної системи. Документування розроблених вимог та моделей для створення інформаційної системи. Управління змінами до вимог протягом циклу розробки програмного забезпечення.

Метою дисципліни є опанування студентами теоретичних та практичних знань і навичок систем тестування, інструментальних засобів розробки програмних систем, розгортання та функціонування програмного забезпечення, розробляти, аналізувати та застосовувати ефективні алгоритми для розв'язання професійних завдань в області інформаційних технологій, та узгодження розробки й постачання програмного забезпечення із його використанням

Предмет дисципліни - вивчення принципів побудови, архітектури, основних функцій, режимів роботи та елементів і комплексів практик систем керування версіями., розроблення адаптивних алгоритмів розв'язування задач.

Завдання. В результаті вивчення дисципліни у студентів повинні сформуватися наступні компетентності:

Вивчення дисципліни «Системи керування версіями» сприяє формуванню у студентів фахової компетентності (ФК) за освітньою програмою:

ФК01 - Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення.

ФК08 - Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення.

Вивчення дисципліни «Системи керування версіями» сприяє формуванню у студентів наступних програмних результатів навчання (ПРН) за освітньою програмою:

ПРН01 - Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.

ПРН03 - Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.

ПРН05 - Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.

ПРН07 - Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.

ПРН09 - Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення.

ПРН14 - Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.

ПРН18 - Знати та вміти застосовувати інформаційні технології обробки, зберігання та передачі даних.

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити дисципліни. Знання, отримані при вивченні дисциплін: "Компоненти програмної інженерії" та "Основи програмування".

Постреквізити дисципліни. Отримані знання при вивченні дисципліни «Компоненти програмної інженерії» формує базові знання для вивчення наступних дисциплін: "Теорія ймовірностей та математична статистика", "Алгоритми та структури даних", "Бази даних", "Моделювання та аналіз програмного забезпечення", "Основи розробки трансляторів", "Системи штучного інтелекту", "Організація баз даних та знань", "Криптографія та шифрування". Також викладений матеріал може бути використаний при вивченні дисциплін "Математичне моделювання систем і процесів", "Дослідження операцій", які викладаються в наступних семестрах. Компетенції, отримані студентами в процесі вивчення цієї дисципліни, використовуються ними при виконанні дипломної роботи.

3. Зміст навчальної дисципліни

Тема 1. Вступ до Git.
Тема 2. Робочий процес та інтеграція з GitHub.
Тема 3. Робоча директорія, Аналіз історії та зроблених змін.
Тема 4. Скасування змін у робочій директорії. Скасування комітів. Зміна останнього коміту.
Тема 5. Індекс. Переміщення по історії.
Тема 6. Розуміння Git. Ігнорування файлів.
Тема 7. Stash. Відкриті проєкти

4. Навчальні матеріали та ресурси

Основна література

1. Інженерія програмного забезпечення: Посібник для студентів вищих навчальних закладів / І. Л. Бородкіна, Г. О. Бородкін ; М-во освіти і науки України, Національний університет біоресурсів та природокористування України. – Київ: , 2018. – с.

2. Підручник з робочого столу GitHub - Співпрацюйте з GitHub зі свого робочого столу - Інший. Огляду Ігри, Розваги, Листопад 2023. URL: https.7/uk.myservername.com/github-desktop-tutorial-collaborate-with-gitliub-from-vour-desktop.
3. Sommerville I. Software Engineering— 9th ed. / Ian Sommerville. – Addison-Wesley, 2011.
4. Що таке Git та Github - керівництво для початківців. SebWeo. URL: https://sebweo.com/scho-take-git-ta-github-kerivnitstvo-dlya-pochatkivtsiv/.
5. The Git experience in Visual Studio. Microsoft Learn: Build skills that open doors in your career. URL: https://learn.microsoft.com/en-us/visualstudio/version-control/git-with-visual-studio?view=vs-2023.
6. Laplante, Phillip (2007). What Every Engineer Should Know about Software Engineering. Boca Raton: CRC. ISBN 9780849372285. Процитовано 2011-01-21.
7. Continuous Delivery: reliable software releases through build, test, and deployment automation. / Humble, Jez; Farley, David (2011). / Pearson Education Inc. ISBN 978-0-321-60191-9.

Додаткова література

1. Програмна інженерія. Візуальне моделювання програмних систем: підручник для СПО / Е. А. Черткова. — 2-е вид., випр. та доп. — М.: Видавництво Юрайт, 2017. — 164 с. ISBN 978-5-534-04928-2
2. 2. Лавріщева Є. М. Програмна інженерія. Парадигми, технології та case-засоби 2-ге вид. Видавництво: Юрайт : ISBN: 5534010568 ISBN-13(EAN):785534010565 Сторінки: 280
3. 3. Основи інженерії якості програмних систем.-Андон Ф. І., Коваль Г. І., Коротун Т. М., Лавріщева О. М., Суслов В. Ю.-Київ: Академперіодика, 2007.-680 с.
4. How to Git(Hub) Sponge 8.0.0. Sponge Documentation. URL: https://docs.spongepowered.org/stable/ru/contributing/howtogit.html.
5. Визначення предмету-програмна інженерія.-/Лавріщева К. М.-Проблеми програмування.-Спецвипуск.-2008.-№ 2-3.-с.191-204.
6. Становлення та розвиток модульно-компонентної інженерії програмування в Україні / Лавріщева О. М.-Препринт 2008-1.-Ін-т кібернетики ім. В. М. Глушкова, 33 с
7. Методи програмування. Теорія, практика, інженерія. -/Лавріщева Е. М.-Наукова думка. -2006.-471с.
8. Peter, Naur; Brian Randell (7–11 October 1968). Software Engineering: Report of a conference sponsored by the NATO Science Committee (PDF). Garmisch, Germany: Scientific Affairs Division, NATO.
9. SWEBOK executive editors, Alain Abran, James W. Moore; editors, Pierre Bourque, Robert Dupuis. (2004). У Pierre Bourque and Robert Dupuis. Guide to the Software Engineering Body of Knowledge - 2004 Version. IEEE Computer Society. с. 1–1. ISBN 0-7695-2330-7.
10. What Is Git | Explore A Distributed Version Control Tool | Edureka. Edureka. URL: https:Z/www.edureka.co biog what-is-git.
11. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations / Gene Kim, Patrick Debois, John Willis, Jez Humble, John Allspaw / IT Revolution Press / October 6, 2016, 480p, ISBN-10 : 1942788002 ISBN-13 : 978-1942788003

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Тема 1. Вступ до Git.
Лекція 1. Вступ.
Історія Git, сучасні тенденції розвитку..
Лекція 2. Масштабування процесу безперервного розгортання.
Розширення повноважень команди. Зменшення кількості помилок. Гнучкість розгортання.
Реліз-кандидат. Принципи розгортання ПО. Повторюваний процес розгортання. Контроль за допомогою системи управління версіями. Безперервне поліпшення.

Тема 2. Робочий процес та інтеграція з GitHub.
Лекція 3. Управління версіями.

Що таке розподілена система керування версіями. Безперервна інтеграція в потокових системах управління версіями. Внесення складних змін без розгалуження. Гілка для випуску. Розгалуження по функціональним засобам. Розгалуження по командам.

Лекція 4. Розподілені системи управління версіями в корпоративних середовищах.

Використання розподілених систем керування версіями. Git. Розгалуження і злиття. Злиття. Гілки в системі безперервної інтеграції. Потокові системи управління версіями. Що таке потокова система управління версіями. Потокові моделі розробки. Статичні і динамічні уявлення.

Тема 3. Робоча директорія, Аналіз історії та зроблених змін.

Лекція 5. Безперервна інтеграція.

Управління середовищем розробки. Використання програм безперервної інтеграції. Розробка через тестування. Пропоновані методики.

Лекція 6. Альтернативні підходи до безперервної інтеграції.

Базова система безперервної інтеграції. Екстремальне програмування. Технічні проблеми. Альтернативні підходи.

Тема 4. Скасування змін у робочій директорії. Скасування комітів. Зміна останнього коміту.

Лекція 7. Реалізація стратегії тестування.

Тести, орієнтовані на ділову логіку і підтримують процес розробки. Тести, орієнтовані на технологію і підтримують процес розробки. Тести, орієнтовані на ділову логіку і критикують проект. Тести, орієнтовані на технологію і критикують проект. Тестові двійники.

Лекція 7. Апробація тестування.

Реальні ситуації і стратегії. Початок проекту. Середина проекту.

Лекція 8. Застарілі системи. Інтеграційне тестування.

Створення тестів. Управління списками невиправлених дефектів.

Тема 5. Індекс. Переміщення по історії.

Лекція 9. Структура конвеєра розгортання.

Базовий конвеєр розгортання. Методики застосування конвеєра розгортання. Рекомендовані методики етапу фіксації.

Лекція 10. Автоматичні приймальні випробування.

Рекомендовані методики автоматичного приймального тестування. Ручне тестування. Тестування не функціональних вимог. Підготовка до випуску. Автоматизація розгортання та постачання релізу. Відкат змін. Стратегія успіху. Реалізація конвеєра розгортання.

Тема 6. Розуміння Git. Ігнорування файлів.

Лекція 11. Сценарії збірки і розгортання.

Огляд інструментів збирання: make, ant, nant і msbuild, maven.

Тема 7. Stash. Відкриті проєкти.

Лекція 12. Розгортання тестових середовищ та їх тестування.

Структура проекту з додатками для jvm / .net. Сценарії розгортання. Шари розгортання і тестування. Тестування конфігурації середовища.

6. Самостійна робота студента

Тема 1. Вступ до Git.

Розгортання шаблонів автоматично та вручну.

Тема 2. Робочий процес та інтеграція з GitHub.

Опанування системи управління версіями Git.

Тема 3. Робоча директорія, Аналіз історії та зроблених змін.
Використання програм безперервної інтеграції, аналіз проблем, опробування альтернативних підходів

Тема 4. Скасування змін у робочій директорії. Скасування комітів. Зміна останнього коміту.
Створення тестів. Апробація.

Тема 5. Індекс. Переміщення по історії.
Автоматизація розгортання та постачання релізу. Реалізація конвеєра розгортання.

Тема 6. Розуміння Git. Ігнорування файлів.
Розгортання тестових середовищ з використанням сценаріїв та інструментів збирання.

Тема 7. Stash. Відкриті проекти.
Моделювання та контроль інфраструктури. Управління інфраструктурою.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Відвідування лекційних та лабораторних занять є обов'язковим за винятком поважних причин (хвороби, форс-мажорних обставин).

В разі пропущення занять з поважних причин викладач надає можливість студенту виконати усі або деякі лабораторні завдання (винятком є виконання деяких завдань у зв'язку із закінченням навчального процесу).

В разі пропущення занять без поважних причин, а також через порушення граничного терміну виконання завдання (deadline) студент може отримати 80% від максимальної оцінки відповідне завдання.

Протягом семестру студенти:

- виконують та захищають лабораторні роботи у відповідні терміни (на кожну лабораторну роботу відводиться два тижні для здачі),
- пишуть модульну контрольну роботу,
- повинні позитивно закрити дві атестації (в кінці березня та в середині травня),
- по закінченні навчального процесу складають екзамен.

8. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Система рейтингових (вагових) балів та критерії оцінювання

1) Робота на лекціях

На лекціях може бути проведено бліцопитування студентів. Такі опитування проводяться на довільних лекціях 5 разів протягом семестру, наприкінці лекції. Ваговий бал за вірну відповідь - 1. Максимальна кількість балів, що може отримати кожен студент за семестр - 5.

2) Лабораторні практикуми

Максимальна кількість балів за усі виконані комп'ютерні практикуми дорівнює 60 балів. Розподіл балів серед лабораторних практикумів наступний:

№ з/п	Назва лабораторного практикуму	Кількість балів
1	Основи роботи з Git та GitHub.	12
2	Інтегрування з хмарними застосунками та сервісами.	12
3	Розгортання в хмарному хостингу.	12
4	Командна робота в GitHub.	12
5	Тестування конфігурації середовища.	12
Всього:		60

Критерії оцінювання:

Виконання лабораторного практикуму:

- виконаний своєчасно (протягом двох тижнів з моменту видачі), у повному обсязі – відповідний бал згідно номеру комп'ютерного практикуму;
- виконаний із запізненням – знімається 10 – 30% від максимальної кількості балів в залежності від терміну запізнення;
- виконаний не самостійно, із запізненням – знімається 50% від максимальної кількості балів;
- невиконани1 протягом відведеного часу – 0 балів.

3) Модульна контрольна робота

Максимальна кількість балів за модульну контрольну роботу дорівнює 10 балів.

Якість виконання роботи:

- усі відповіді вірні та повні –10 балів,
- у відповідях допущені несуттєві неточності – 8 балів,
- половина відповідей вірна – 5 балів,
- відповіді з суттєвими неточностями, але без критичних помилок – 2 бали,
- менше половини відповідей вірна – 0 балів.

Штрафні та заохочувальні бали за:

- активність на комп'ютерних практикумах + 2 бали
- виконання комп'ютерного практикуму з використанням власного оптимального алгоритму + 1 бали
- відсутність на занятті без поважної причини – 2 бали
- несвоєчасна здача комп'ютерного практикуму (пізніше ніж за тиждень) – 0,5 балів;

4) Складання заліку

Максимальний ваговий бал $r_{icп}$=40

На іспиті студент виконує письмову контрольну роботу, яка містить два теоретичних питання і одне практичне питання. Теоретичні питання оцінюються максимально по 10 балів, практичне – 20 балів.

Умови позитивної проміжної атестації

Для отримання „зараховано" з першої проміжної атестації студент матиме не менше ніж 30 балів (за умови, що за 8 тижнів згідно з календарним планом контрольних заходів „ідеальний" студент має отримати 12 +12 +12 = 36 бал).

Для отримання „зараховано" з другої проміжної атестації студент матиме не менше ніж 50 балів (за умови, що за 14 тижнів згідно з календарним планом контрольних заходів „ідеальний" студент має отримати 36 +12 +12 =60 балів).

Умови допуску до *залікуу*

Необхідною умовою допуску до заліку є зарахування усіх комп'ютерних практикумів та виконання модульної контрольної роботи, а також стартовий рейтинг (R_c) не менше 40 балів.

Розрахунок шкали (R) рейтингу:

Сума вагових балів контрольних заходів протягом семестру (шкала рейтингу) складає:

$$R = r_{лек} + r_{практ} + r_{мод} + r_{icп} = 5+45 + 10 + 40= 100 \text{ балів.}$$

Максимальний стартовий рейтинг становить $R_c = r_{лек} + r_{практ} + r_{мод} = 60$ балів.

Рейтинг заліку дорівнює 40 балів. Мінімальний рейтинг допуску до заліку становить 40 балів.

Таким чином, рейтингова шкала з кредитного модуля складає

$$R = 60+40=100 \text{ балів.}$$

Для отримання студентом відповідних оцінок рейтингова оцінка студента переводиться згідно таблиці:

Бали	Оцінка
95 - 100	Відмінно
85 - 94	Дуже добре
75 - 84	Добре
65 - 74	Задовільно
60 - 64	Достатньо
Менше 60	Незадовільно
R < 40 є незараховані роботи комп'ютерного практикуму або не виконані інші умови допуску до екзамену	Не допущено

9. Додаткова інформація з дисципліни (освітнього компонента)

Перелік питань, які виносяться на семестровий контроль:
1. Основи роботи з Git та GitHub.
2. Використання розподілених систем керування версіями. Git.
3. Безперервне поліпшення. Зменшення кількості помилок.
4. Розширення повноважень команди. Зменшення кількості помилок.
5. Гнучкість розгортання. Реліз-кандидат. Принципи розгортання ПО.
6. Контроль за допомогою системи управління версіями. Безперервне поліпшення.
7. Git. Управління версіями.
8. Безперервна інтеграція в потокових системах управління версіями. Внесення складних змін без розгалуження.
9. Гілка для випуску. Розгалуження по функціональним засобам. Розгалуження по командам.
10. Розподілені системи управління версіями в корпоративних середовищах.
11. Що таке потокова система управління версіями. Потокові моделі розробки. Статичні і динамічні уявлення.
12. Масштабування процесу безперервного розгортання. Безперервна інтеграція. Управління середовищем розробки.
13. Використання програм безперервної інтеграції. Розробка через тестування. Пропоновані методики.
14. Альтернативні підходи до безперервної інтеграції.
15. Базова система безперервної інтеграції. Екстремальне програмування.
16. Технічні проблеми. Альтернативні підходи.
17. Реалізація стратегії тестування.
18. Тести, орієнтовані на ділову логіку і підтримують процес розробки.
19. Тести, орієнтовані на технологію і підтримують процес розробки.
20. Тести, орієнтовані на ділову логіку і критикують проект.
21. Тести, орієнтовані на технологію і критикують проект. Тестові двійники.
22. Апробація тестування.
23. Реальні ситуації і стратегії. Початок проекту. Середина проекту.
24. Застарілі системи. Інтеграційне тестування.
25. Створення тестів. Управління списками невиправлених дефектів.
26. Конвеєр розгортання.
27. Структура конвеєра розгортання.
28. Базовий конвеєр розгортання.
29. Методики застосування конвеєра розгортання.
30. Рекомендовані методики етапу фіксації.
31. Автоматичні приймальні випробування.
32. Рекомендовані методики автоматичного приймального тестування. Ручне тестування.
33. Тестування не функціональних вимог. Підготовка до випуску.

34. Автоматизація розгортання та постачання релізу. Відкат змін.
35. Стратегія успіху. Реалізація конвеєра розгортання.
36. Принципи створення сценаріїв збірки і розгортання.
37. Сценарії збірки і розгортання.
38. Інструменти збирання: make, ant, nant і msbuild, maven.
39. Розгортання тестових середовищ та їх тестування.
40. Структура проекту з додатками для jvm / .net.
41. Сценарії розгортання. Шари розгортання і тестування.
42. Тестування конфігурації середовища.

Складено старшим викладачем ІПЗЕ, Колумбетом Вадимом Петровичем

Ухвалено кафедрою інженерії програмного забезпечення в енергетиці НН ІАТЕ (протокол № 34 від 10.05.2024 р.)
Погоджено Методичною комісією НН ІАТЕ (протокол № 9 від 31.05.2024 р.)