МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗАТВЕРДЖУЮ
В. о. декана факультету
комп'ютерних інформаційних
технологій
_____ Ігор ЯКИМЕНКО
"__" _____2023 р.

ЗАТВЕРДЖУЮ
В. о. проректора
з науково-педагогічної роботи
_____ Віктор ОСТРОВЕРХОВ
"__" _____2023 р.

ЗАТВЕРДЖУЮ
Директор навчально наукового
інституту новітніх освітніх
технологій
_____ Святослав ПИТЕЛЬ
"__" _____2023 р.

РОБОЧА ПРОГРАМА
з дисципліни
«Конструювання програмного забезпечення»

Ступінь вищої освіти: бакалавр
Галузь знань – 12 Інформаційні технології
Спеціальність – 121 Інженерія програмного забезпечення
Освітньо-професійна програма – «Інженерія програмного забезпечення»

Кафедра комп'ютерних наук

Форма навчання	Курс	Семестр	Лекції (год.)	Лаб. (год.)	ІРС (год.)	Тренінг (год.)	СРС (год.)	Разом (год.)	Екз. (сем.)
Денна	3	5	28	28	3	8	113	180	5
Заочна	3	5, 6	8	4	-	-	168	180	6

31.08.2023

Тернопіль – ЗУНУ
2023

Робоча програма складена на основі освітньо-професійної програми підготовки бакалаврів галузі знань 12 «Інформаційні технології» спеціальності 121 «Інженерія програмного забезпечення», затвердженої Вченою радою ЗУНУ №_9_ від _26.05_____2024 р.

Робоча програма розроблена доцентом кафедри комп'ютерних наук, к.т.н. Володимиром МАНЖУЛОЮ

Робоча програма затверджена на засіданні кафедри комп'ютерних наук, протокол №1 від 28 серпня 2023р.

Завідувач кафедри д.т.н, професор _____Андрій ПУКАС

Розглянуто та схвалено групою забезпечення спеціальності 121 Інженерія програмного забезпечення, протокол №_1_ від _30.08_____ 2023р.

Голова групи
забезпечення спеціальності,
д.т.н., професор _____Микола ДИВАК

Гарант ОП,
к.т.н., доцент _____ Світлана КРЕПИЧ

СТРУКТУРА РОБОЧОЇ ПРОГРАМИ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«Конструювання програмного забезпечення»

1. Опис дисципліни «Конструювання програмного забезпечення»

Дисципліна – «Конструювання програмного забезпечення»	Галузь знань, спеціальність, ступінь вищої освіти	Характеристика навчальної дисципліни
Кількість кредитів ECTS- 6	Галузь знань: 12 Інформаційні технології	**Статус дисципліни**: нормативна **Мова навчання**: українська
Кількість залікових модулів – 4	Спеціальність: 121 Інженерія програмного забезпечення	Рік підготовки: *Денна – 3* *Заочна – 3* Семестр: *Денна – 5* *Заочна -5, 6*
Кількість змістових модулів – 3	Ступінь вищої освіти: бакалавр	Лекції: *Денна – 28 год.* *Заочна – 8 год.* Лабораторні заняття: *Денна – 28 год.* *Заочна – 4 год.*
Загальна кількість годин – 180		Самостійна робота: *Денна – 113 год.* *Тренінг – 8* *Заочна – 168 год.* Індивідуальна робота: *Денна* – 3 год.
Тижневих годин: Денна форма навчання 5 семестр – 12 год., з них аудиторних – 4 год.		Вид підсумкового контролю: *Денна* – екзамен, 5; *Заочна* – екзамен, 6.

2. Мета й завдання вивчення дисципліни «Конструювання програмного забезпечення»

2.1. Мета вивчення дисципліни

Метою дисципліни «Конструювання програмного забезпечення» є опанування невід'ємним етапом процесу створення ПЗ, а саме конструювання коду. Усі інші етапи створення ПЗ, такі як аналіз і проектування також є потрібними і важливими, але їх можливо провести за скороченою програмою чи неформально, подумки, тоді як створити виконуваний комп'ютером програмний продукт оминаючи процес конструювання – неможливо. Тобто в основному увага дисципліни зосереджена на програмному коді, мінімізації його складності, а отже й кількості помилок у ньому.

Метою дисципліни «Конструювання програмного забезпечення» є створення якісного коду ПЗ із залученням команди розробників за допомогою

розподіленої системи контролю версій, технік відлагоджування, зразків детального проектування та юніт-тестування.

Метою проведення лекцій є формування у студентів системи теоретичних знань з курсу «Конструювання програмного забезпечення».

Мета проведення лабораторних занять полягає у набутті студентами практичних навиків конструювання ПЗ, написання юніт-тестів, роботи у команді програмістів засобами розподіленої системи контролю версій.

2.2. Завдання вивчення дисципліни

Лекційні заняття курсу «Конструювання програмного забезпечення» орієнтовані на набуття студентами теоретичних знань з:

- понять cohesion та coupling;
- якісного оцінювання складності коду;
- теорії контрою версій коду;
- створення вдалих юніт-тестів;
- технік відлагоджування коду на основі точок зупину (break-points), трейсів стеку (stack-traces), протоколювання (logging);
- важливих засобів рефакторингу і застосування зразків проектування для мінімізації складності програмного коду;

Завдання проведення лабораторних занять:

- навчитися працювати із засобами рефакторингу інтегрованих середовищ розробки коду;
- навчитися працювати з відлагоджувальником в інтегрованих середовищах розробки;
- навчитися працювати з консольними командами системи розподіленого контролю версій та, факультативно, будь-яким графічним її фронт-ендом;
- навчитися писати юніт-тести на існуючий та новий функціонал;
- навчитися ефективно застосовувати засоби протоколювання та обробки виключних ситуацій.

2.3 Найменування та опис компетентностей, формування котрих забезпечує вивчення дисципліни «Конструювання програмного забезпечення»:

- здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу.
- здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.
- здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки-цілісності, загальної функціональності і надійності програмного забезпечення.

2.4. Передумови для вивчення дисципліни.

Дисципліни, які повинні бути вивчені попередньо:

- «Основи інженерії програмного забезпечення»;
- «Об’єктно-орієнтоване програмування»;
- «Аналіз вимог до програмного забезпечення»;

-

2.5. Результати навчання

У результаті вивчення курсу «Конструювання програмного забезпечення» студенти повинні:

РН02. Знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії програмного забезпечення і дотримуватись їх в професійній діяльності.

РН03. Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.

РН04. Знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії програмного забезпечення.

РН06. Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.

РН15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.

РН16. Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації

РН22. Знати та вміти застосовувати методи та засоби управління проектами.

3. Зміст дисципліни «Конструювання програмного забезпечення»

Змістовий модуль 1. Вступ в конструювання ПЗ

Тема 1. Місце конструювання в ЖЦ розробки ПЗ. Огляд засобів конструювання

Поняття конструювання ПЗ. Поняття cohesion та coupling.

Література:

[1 - 3] — розділи 2, 3.

Тема 2. Структурування і форматування коду. Ієрархія пакетів застосунку (model, ui, util). Конвенції щодо форматування. Javadoc коментарі

Огляд документів Microsoft та Oracle, що описують ковенції форматування коду для мов програмування C# та Java. Правила ієрархії простору імен чи пакетів.

Література:

[1] — розділи 31, 32.

C# Coding Conventions (C# Programming Guide) – https://msdn.microsoft.com/en-us/library/ff926074.aspx.

Тема 3. Налаштування на автора та створення локальних репозиторіїв в системі розподіленого контролю версій git

Налаштування розподіленої системи контролю версій git в ОС Linux та Windows: вказання автора, його адреси електронної пошти. Створення репозиторію за допомогою команди git init.

Література:

[Chacon S.] – стор. 1-31.

[Chacon S.] – стор. 33-35.

Тема 4. Клонування репозиторію з сервера і публікування репозиторію на сервері. Поняття локальної і віддаленої гілки

Огляд команд clone, push, branch.

Література:

[Chacon S.] – стор. 36-63.

Тема 5. Створення і підтримування структури гілок. Об'єднання (merge) гілок і перебазування (rebase) на основі гілок.

Створення гілки за допомогою команди checkout. Усунення змін за допомогою команди checkout. Об'єднання гілок за допомогою команди merge. Перебазування гілок за допомогою команди rebase. Визначення випадків у котрих краще застосувати merge або rebase.

Література:

[Chacon S.] – стор. 79-88.

[Chacon S.] – стор. 89-113.

Змістовий модуль 2. Модульне тестування

Тема 6. Модульне тестування

Мета і правила розробки Unit-тестів. Створення Unit-тестів на основі JUnit

Література:

[Meszaros G.] – стор. 3-30.

Тема 7. Роль ін'єкції залежностей (dependency injection) для зменшення складності конструювання і тестування ПЗ

Література:

[Seemann M.] – стор. 3-28.

Тема 8. Імітування залежностей на основі Mockito

Література:

http://mockito.org/

Тема 9. Автоматизоване тестування ui на основі com.robotium.solo

Література:

https://github.com/RobotiumTech/robotium

Тема 10. Інтеграційне тестування на основі http://testfairy.com/

Тестування і відлагоджування.

Література:

[МакКоннелл] — розділи 22, 23.

Змістовий модуль 3. Просунуті питання констрування ПЗ

Тема 11. Розв'язування конфліктів при контролі версій

Література:

[Chacon S.] – стор. 302-321.

Тема 12. Об'єднання комітів засобами git rebase -i HEAD~N

Література:

[Chacon S.] – стор. 272-290.

Тема 13. Пошук у історії комітів за автором чи зразком тексту

Література:

[Chacon S.] – стор. 44-55.

Тема 14. Конструювання для повторного використання

Питання рефакторингу для полегшення повторного використання.

Література:

[МакКоннелл] — розділи 24-26;

[Fowler M.] – стор. 13-137; стор. 311-327.

4. Структура залікового кредиту з дисципліни «Конструювання програмного забезпечення»

(денна форма навчання)						
	Кількість годин					
	Лекції	Лабораторні заняття	Самостійна робота	ІРС	Тренінг КПІЗ	Контрольні заходи
Змістовий модуль 1. Вступ в конструювання ПЗ						
Тема 1. Місце конструювання в ЖЦ розробки ПЗ. Огляд засобів конструювання	2		8	1	3	Усне опитування та тестування
Тема 2. Структурування і форматування коду. Ієрархія пакетів застосунку (model, ui, util). Конвенції щодо форматування. Javadoc коментарі	2	2	8			Усне опитування та тестування
Тема 3. Налаштування на автора та створення локальних репозиторіїв в системі розподіленого контролю версій git	2	2	8			Усне опитування та тестування
Тема 4. Клонування репозиторію з сервера і публікування репозиторію на сервері. Поняття локальної і віддаленої гілки	2	2	8			Усне опитування та тестування
Тема 5. Створення і підтримування структури гілок. Об’єднання (merge) гілок і перебазування (rebase) на основі гілок.	2	2	8			Усне опитування та тестування
Змістовий модуль 2. Робота з Unit-тестами						
Тема 6. Мета і правила розробки Unit-тестів. Створення Unit-тестів на основі JUnit	2	2	8	1	3	Усне опитування та тестування
Тема 7. Роль ін’єкції залежностей (dependency injection) для зменшення складності конструювання і тестування ПЗ	2	2	8			Усне опитування та тестування
Тема 8. Імітування залежностей на основі Mockito	2	2	8			Усне опитування та тестування
Тема 9. Автоматизоване тестування ui на основі com.robotium.solo	2	2	8			Усне опитування та тестування
Тема 10. Інтеграційне тестування на основі http://testfairy.com/	2	2	8			Усне опитування та тестування

Змістовий модуль 3. Просунуті питання конструювання ПЗ						
Тема 11. Розв’язування конфліктів при контролі версій	2	2	8	1	2	Усне опитування та тестування
Тема 12. Об’єднання комітів засобами git rebase -i HEAD~N	2	2	8			Усне опитування та тестування
Тема 13. Пошук у історії комітів за автором чи зразком тексту	2	2	8			Усне опитування та тестування
Тема 14. Конструювання для повторного використання	2	4	8			Усне опитування та тестування
Разом	28	28	113	3	8	

(заочна форма навчання)			
	Кількість годин		
	Лекції	Лабораторні заняття	Самостійна робота
Тема 1. Місце конструювання в ЖЦ розробки ПЗ. Огляд засобів конструювання	1		8
Тема 2. Структурування і форматування коду. Ієрархія пакетів застосунку (model, ui, util). Конвенції щодо форматування. Javadoc коментарі	1	1	10
Тема 3. Налаштування на автора та створення локальних репозиторіїв в системі розподіленого контролю версій git	1		10
Тема 4. Клонування репозиторію з сервера і публікування репозиторію на сервері. Поняття локальної і віддаленої гілки	1	1	10
Тема 5. Створення і підтримування структури гілок. Об’єднання (merge) гілок і перебазування (rebase) на основі гілок.			10
Тема 6. Мета і правила розробки Unit-тестів. Створення Unit-тестів на основі JUnit	1	1	10
Тема 7. Роль ін’єкції залежностей (dependency injection) для зменшення складності конструювання і тестування ПЗ	1		10
Тема 8. Імітування залежностей на основі Mockito			10
Тема 9. Автоматизоване тестування ui на основі com.robotium.solo			10
Тема 10. Інтеграційне тестування на основі http://testfairy.com/			10
Тема 11. Розв’язування конфліктів при контролі версій	1	1	10
Тема 12. Об’єднання комітів засобами git rebase -i HEAD~N			10

Тема 13. Пошук у історії комітів за автором чи зразком тексту			10
Тема 14. Конструювання для повторного використання	1		10
Разом	8	4	138

5. Тематика лабораторних занять

Лабораторне заняття № 1
Тема: Репозиторії git
Мета: Встановлення git в ОС Linux і Windows, Налаштування глобальних параметрів ідентифікації розробника. Створення репозиторію з першим комітом.
Література: 1-4.

Лабораторне заняття № 2
Тема: Дотримування конвенцій кодування. Javadoc
Мета: набуття практичних навичок дотримання конвенцій щодо форматування і документування коду, використання анотацій.
Література: 1-4.

Лабораторне заняття № 3
Тема: Створення Unit-тестів
Мета: набуття практичних навичок роботи з JUnit framework.
Література: 1-4.

Лабораторне заняття № 4
Тема: Написання автоматизованих сценаріїв тестування ui
Мета: навчитися використовувати компоненти com.robotium.solo для тестування ui в ОС Android.
Література: 1-4.

Лабораторне заняття № 5
Тема: Робота з Framework імітування залежностей mockito
Мета: отримати практичні навики роботи з фреймворком імітації залежностей mockito.
Література: 1-4.

Лабораторне заняття № 6
Тема: Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів MVC застосунку
Мета: набути практичні навики конструювання MVC застосунків в Spring Framework з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів.

Література: 1-4.

Лабораторне заняття № 7
Тема: Написання Unit-тестів, що перевіряють вірність збереження даних у ORM JPA, Hibernate
Мета: навчитися тестувати JPA, Hibernate ORM для перевірки вірності збереження даних.
Література: 1-4.

6. Комплексне практичне індивідуальне завдання

Індивідуальні завдання з дисципліни «Конструювання програмного забезпечення» виконується самостійно кожним студентом. КПІЗ охоплює усі основні теми дисципліни «Конструювання програмного забезпечення». Метою виконання КПІЗ є оволодіння навичками застосування теоретичних занять. КПІЗ оформлюється у відповідності з встановленими вимогами. Виконання КПІЗ є однією із обов'язкових складових модулів залікового кредиту з дисципліни «Конструювання програмного забезпечення».

Варіанти КПІЗ з дисципліни «Конструювання програмного забезпечення»:

1. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-front-end нагромадження і коментування фотографій для ОС Android.

2. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-front-end "консольний чат" на мові програмування Java.

3. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-front-end "консольний чат" на мові програмування Scala.

4. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-front-end "Чат для ОС Android".

5. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-back-end "REST чат" засобами Spring Framework.

6. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-back-end "REST сховище фотографій" засобами Spring Framework.

7. Конструювання з використанням розподіленої системи контрою версій, рефакторинг, відлагоджування і написання unit-тестів програми-front-end нагромадження і коментування відео для ОС Android.

7. Самостійна робота

1. Життєвий цикл програмного забезпечення.
2. Різновиди моделей життєвого циклу програмного забезпечення.

3. Основні характеристики і фази водоспадної моделі.
4. Основні характеристики і фази ітеративної моделі.
5. Основні характеристики і фази спіральної моделі.
6. Основні аспекти планування проекту програмного забезпечення.
7. Основні характеристики планування процесу розробки.
8. Основні характеристики визначення результатів проекту.
9. Основні характеристики оцінки зусиль, розкладу та вартісних очікувань проекту.
10. Основні характеристики розподілу ресурсів проекту.
11. Критичний шлях проекту.
12. Основні характеристики керування ризиками проекту.
13. Основні типи мов конструювання програмних проектів.
14. Основні можливості конфігураційних мов конструювання.
15. Основні можливості інструментальних мов конструювання.
16. Основні можливості мов програмування.
17. Методи інтеграції.
18.Засоби безперервної інтеграції.
20. 23. Цілі тестування програмного забезпечення.
21. Техніки що базуються на інтуїції та досвіді інженера.
22. Техніки що базуються на специфікаціях.
23. Техніки що орієнтовані на код.
24. Техніки що орієнтовані на дефекти.
25. Техніки що базуються на умовах використання.
26. Техніки що базуються на природі застосування.
37. Методи оцінки результатів тестування.
38. Основні техніки відлагодження програмного забезпечення.
39. Системи протоколювання, шо реалізують рівні сповіщення debug, error, warn, info та ін.;
40. Дослідження системи сповіщення про креші Crashlytics;
41. Рефакторинг для покращення організації даних (розліл 8 книги [Fowler M.]);
42. Рефакторинг для спрощення умовних виразів (розліл 9 книги [Fowler M.]);
43. Рефакторинг для спрощення виклику методів (розліл 10 книги [Fowler M.]);
44. Рефакторинг на основі узагальнень (generalization) (розліл 11 книги [Fowler M.]);
45. Крупномасштабний рефакторинг (розліл 12 книги [Fowler M.]);
46. Засоби рефакторингу (розліл 14 книги [Fowler M.]);
47. Зразки подвійного тестування (розділ 23 книги [Meszaros G.]);
48. Зразки організації тестів (розділ 24 книги [Meszaros G.]);
49. Зразки тестування коду, що працює з базами даних (розділ 25 книги [Meszaros G.]);
50. Зразки проектування, що полегшують тестування (розділ 26 книги [Meszaros G.]);

8. Організація і проведення тренінгу

Тематика: Конструювання ПЗ.

Порядок проведення:

1. Етап детального проектування.
2. Написання функціональної частини застосунку.
3. Відлагодження застосунку.
4. Створення та підримка репозиторію проекту.
5. Написання Unit-тестів до функціональної частини.
6. Написання ui частини застосунку.
7. Написання Unit-тестів до ui частини.
8. Імплементувати методи класів.

9. Методи оцінювання

У процесі вивчення дисципліни «Конструювання програмного забезпечення» використовуються наступні методи оцінювання навчальної роботи студентів:

- поточне опитування;
- стандартизовані тести;
- залікове модульне тестування та опитування;
- презентації результатів виконання завдань;
- оцінювання результатів КПІЗ;
- ректорська контрольна робота;
- тренінг;
- екзамен.

10. Критерії, форми поточного та підсумкового контролю

Підсумковий бал (за 100-бальної шкалою) з дисципліни «Конструювання програмного забезпечення» визначається як середньозважена величина, залежно від питомої ваги кожної складової залікового кредиту:

Заліковий модуль 1	Заліковий модуль 2 (РКР)	Заліковий модуль 3 (КПІЗ)	Заліковий модуль 4 (екзамен)	Разом
20%	20 %	20 %	40%	100 %
1. Усне або письмове опитування під час заняття – 40 балів. 2. Лабораторні роботи – 60 балів	1.Тестові завдання (30 тестів по 2 бали за тест) – 60 балів 2. Завдання 1 – 20 балів 3. Завдання 2 – 20 балів	1. Виконання та захист КПІЗ – 60 балів Лабораторні роботи – 20 балів 2. Виконання завдань під час тренінгу – 20 балів	1.Тестові завдання (30 тестів по 2 бали за тест) – 60 балів 2. Завдання 1 – 20 балів 3. Завдання 2 – 20 балів	100

Шкала оцінювання:

За шкалою ЗУНУ	За національною шкалою	За шкалою ECTS
90-100	відмінно	A (відмінно)
85-89	добре	B (дуже добре)
75-84		C (добре)
65-74	задовільно	D (задовільно)
60-64		E (достатньо)
35-59	незадовільно	FX (незадовільно з можливістю повторного складання)
1-34		F (незадовільно з обов'язковим повторним курсом)

11. Інструменти, обладнання та програмне забезпечення, використання яких передбачає навчальна дисципліна

№	Найменування	Номер теми
1.	Мультимедійний проектор	1-14
2.	Проекційний екран	1-14
3.	Комунікаційне програмне забезпечення (Internet Explorer, Google Chrome, Firefox)	1-14
4.	Операційна система Windows, наявність доступу до мережі Internet	1-14
5.	Персональні комп’ютери	1-14
6.	Комунікаційне програмне забезпечення (Zoom) для проведення занять у режимі онлайн (за необхідності)	1-14
7.	Комунікаційна навчальна платформа (Moodle) для організації дистанційного навчання (за необхідності)	1-14
8.	Базове програмне забезпечення Microsoft Office	1-14
9.	Спеціалізоване програмне забезпечення: - Microsoft Visual Studio, - Mockito - com.robotium.solo	2-14

Рекомендовані джерела інформації

1. McConnell Steve. Code Complete: A Practical Handbook of Software Construction, Second Edition / Steve McConnell. – Publisher: Microsoft Press; 2nd edition, 2019. – 960 p.

2. Nuradil Alymkulov, Diana Ruslanova. Code Refactoring: Meaning, Benefits and Best Practices

3. Vladimir Khorikov. Unit Testing Principles, Practices, and Patterns. – Manning. – Jan 6, 2020

4. Molyneaux I. The Art of Application Performance Testing. Help for Programmers and Quality Assurance. – Publisher: O'Reilly Media, 2019. – 158 p.

5. Stephen Rylander. Patterns of Software Construction: How to Predictably Build Results. – Kindle Edition. Publisher: Apress (February 28, 2023). – 190 p.

6. Martin Fowler. Refactoring: Improving the Design of Existing Code (2nd Edition). – Addison-Wesley Professional. – 2019. 424 p.

7. Refactoring development tool: Visual studio intellicode https://visualstudio.microsoft.com/de/services/intellicode/

8. Refactoring development tool: The IDE for pure Java and Kotlin development. https://www.jetbrains.com/idea/download/

9. C# Coding Conventions (C# Programming Guide) – https://msdn.microsoft.com/en-us/library/ff926074.aspx.

10. Code Conventions for the Java TM Programming Language – http://www.oracle.com/technetwork/java/codeconvtoc-136057.html

11. Package by feature, not layer – http://www.javapractices.com/topic/TopicAction.do?Id=205