$$O\left(H\middle|\bigcap_{j=1}^{J}\bar{E}_j\right)=O(H)\prod_{j=1}^{J}N_{E_j}=0.025$$

Отже, можна зробити висновок, що при наявності всіх симптомів у пацієнта можна поставити діагноз Covid-19, так як апостеріорні шанси (ймовірність) є достатньо високими. У випадку відсутності усіх наведених симптомів у розрахунках, можна визначити, що пацієнт не має такого діагнозу як Covid-19, тому що апостеріорні шанси (ймовірність) є дуже малими значеннями. Отримані дані можна застосувати у подальшій розробці діагностуючої системи для практичного використання у відповідних сферах.

Список літератури

1. Набір даних COVID-19 Symptoms Checker [Електронний ресурс]. Режим доступу – https://www.kaggle.com/datasets/iamhungundji/covid19-symptoms-checker?resource=download.

2. Т.В. Нескородєва, Є.Є. Федоров, Т.В. Січко, А.Р. Нескородєва. Експертні та рекомендаційні системи: навч. посіб. Вінниця: ДонНУ імені Василя Стуса, 2022. 208 с.

3. Зінченко Б.В., Нескородєва Т.В., Аналіз даних про короновірус у світі методами статистичного навчання. // Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" - Вінниця: ДонНУ імені Василя Стуса, 2020, с. 19-22. URL: https://jait.donnu.edu.ua/article/view/8888

УДК 004.4'23

Васильченко Д.Н., студент 4 курсу спеціальності 122 «Комп'ютерні науки»
Опанасюк Б.М., студент 4 курсу спеціальності 122 «Комп'ютерні науки»
Нескородєва Тетяна Василівна, д.т.н., доцент, завідувач кафедри інформаційних технологій

ДОСЛІДЖЕННЯ ЗАСОБУ АВТОМАТИЗОВАНОЇ РОЗРОБКИ APACHE MAVEN

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ

Для сучасних розробників програмного забезпечення завжди гостро стоїть питання автоматизації різноманітних програмних завдань, які вони мають вирішувати у повсякденних робочих процесах, проблемах. Одним із найбільш передових і розповсюджених серед спеціалістів серверних технологій можна сміливо назвати програмне рішення Maven від Apache Group [1].

Apache Maven – потужний інструмент, що надає набір можливостей для спрощеної збірки, публікації та розгортання найрізноманітніших сукупностей проектів одночасно для якнайбільш продуктивного проджект-менеджменту; він дозволяє розробникам реалізовувати найскладніші архітектурні підходи та

документувати повний цикл розробки програмного продукту. Питання оптимізації проєктів розглянути в роботах [2,3].

Maven написаний на Java та використовується для створення проектів, написаних на C#, Scala, Ruby та інших сучасних мов програмування. Цей інструмент вже після найпершого свого релізу полегшив життя розробників Java як під час розробки звітів так і для перевірки побудови та автоматизації тестування.

Загалом, мета Maven — це надання розробникам:

- комплексної та зручної у використанні одночасно багаторазової та простої моделі для написання проектів різної складності.

- набору інструментів і плагінів, які дають змогу взаємодіяти з декларативною моделлю.

Основна частина

Отже, перед тим як описувати принципи роботи засобу Apache Maven, звернемо увагу на те, які проблемні питання він ставить собі за мету вирішити. До них відносяться:

1. Полегшення самої розробки додатку.

Використання Maven, звісно, передбачає розуміння основного функціоналу, але дозволяє уникати вивчення та використання коду більш низького рівня за допомогою вже реалізованих алгоритмів.

2. Створення єдиного середовища складання (побудови) розробки.

Maven складає проект за допомогою єдиної об’єктної моделі проекту, так званої POM, та ряду плагінів. Це надає уніфікації цьому програмному засобу, тобто розуміючи як працює одна програма розроблена за допомогою Apache Maven, вам будуть зрозумілі і будь-які інші.

3. Надання широкого спектру технічної інформації про проект.

Серед тих даних, що надає Maven, використовуючи вже згаданий POM, а також саме джерела проекту, головними є журнал внесених змін в розробку, перехресні посилання на джерела, розсилочні списки джерел програми, використовувані залежності, тобто посилання на бібліотеки з корисним для конкретної роботи кодом і тестувальні звіти.

4. Стимулювання кращих практик роботи з програмним кодом.

Маючи широку базу найкращих практик принципів розробки програмного забезпечення, Maven здатен пропонувати вам визначити структуру каталогів проекту, вказувати на певні проблеми ваших програмних рішень у результаті виконання тестів над ними тощо.

Найчастіше Maven використовується для проектів побудованих на Java, допомагаючи підвантажувати залежності, які стосуються сторонніх бібліотек або JAR-файлів. Інструмент допомагає отримати потрібні файли JAR для кожного проекту, оскільки кожен окремий пакет може містити в собі абсолютно різні версії [4].

Для того щоб розпочати роботу з Maven, не потрібно відвідувати офіційні веб-сайти кожного розробника бібліотек/модулів для завантаження потрібних залежностей — ви просто можете відвідати mvnrepository.com, щоб знайти

необхідні бібліотеки для роботи над вашим проектом. Інструмент також допомагає створити правильну структуру проекту в сервлетах/веб-додатках, налаштувати коректні архетипи тощо.

Тепер варто розглянути життєвий цикл побудови проекту Maven (див. рис. 1).

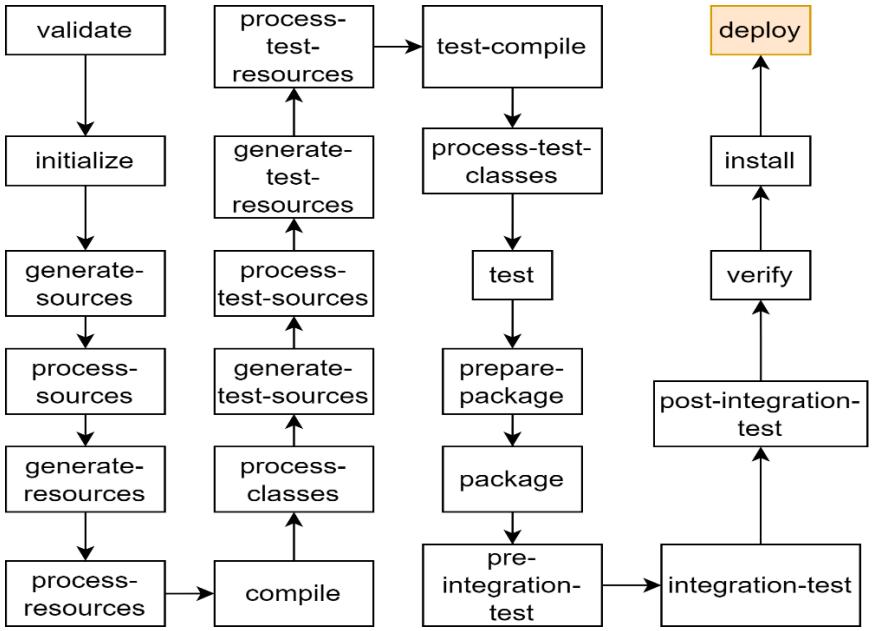

Рисунок 1 – Життєвий цикл Maven проекту

validate: перевірка проекту на правильність усієї необхідної інформації є.

initialize: ініціалізування каталогів збірки.

generate-sources: створення вихідного коду.

process-sources: обробка вихідного коду шляхом розміщення коду в потрібному місці.

generate-resources: створення інформації про ресурси.

process-resources: обробка ресурсів шляхом розміщення їх у правильній директорії.

compile: компіляція вихідного коду проекту.

process-classes: обробка згенерованих скомпільованих класів. Наприклад: покращення байт-коду, обфускація байт-коду тощо.

generate-test-sources: створення вихідного коду тесту.

process-test-sources: обробка вихідного коду тесту.

generate-test-resources: генерація ресурсу для тестування.

process-test-resources: обробка тестового ресурсу, наприклад копіювання його в правильний каталог.

test-compile: компіляція вихідного коду тесту.

process-test-classes: обробка скомпільованого файлу тестового класу.Такі як покращення байт-коду, обфускація коду… тощо.

test: запуск згенерованих тестових випадків за допомогою інфраструктури модульного тестування, наприклад JUnit.

prepare-package: виконання операцій попереднього пакування. Наприкінці цієї фази виводиться розпакована оброблена версія упакування.

package: виконання пакування в розподіленому форматі, наприклад, jar, war або ear.

pre-integration-test: підготовка до інтеграційного тесту.

integration-test: виконання інтеграційного тесту.

post-integration-test: виконання необхідних дій для після інтеграційного тесту. Наприклад, очищення середовища.

verify: перевірка згенерованого розподіленого пакета.

install: встановлення пакету в локальний репозиторій.

deploy: копіювання остаточної збірки у віддалений репозиторій [5].

Висновки

Підсумовуючи все вище описане, ми можемо заявити, що Maven в основному зосереджується на спрощенні та стандартизації процесу збірки проектів та бере на себе відповідальність за:

- збірку проектів;
- документування циклу розробки;
- побудову залежностей та програмних модулів/бібліотек;
- створення звітів;
- відслідковування та управління змінами під час розробки;
- розподілення;
- випуск релізів.

І саме тому найкращими практиками використання Maven Build Tool можна назвати:

- роботу з великою кількістю залежностей - в такому випадку Maven надає зручний інструментарій та допомагає легко впоратися з кожною із них;
- при існуванні реальної потреби своєчасного оновлення версій сторонніх бібліотек потрібно лише оновити ідентифікатор: версії у конфігураційному файлі pom щоб працювати з новітніми версіями залежностей;
- безперервне збирання, інтеграція та тестування можна легко виконати за допомогою Maven [3];
- використання при необхідності знайти простий спосіб генерації документації з вихідного коду, його компіляції та упаковки скомпільованого коду у файли JAR або ZIP.

Список літератури

1. *Maven. Introduction [Електронне видання]. URL: https://maven.apache.org/what-is-maven.html (дата звернення: 03.12.2022).*
2. *Якубич К.О., Нескородєва Т.В. Аналіз та планування проекту «Розробка сайту» засобами програми MS Project. // Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" - Вінниця: ДонНУ імені Василя Стуса, 2022, с. 49-52. URL: https://jait.donnu.edu.ua/article/view/12254*
3. *Новицький М.О., Нескородєва Т.В. Аналіз та оптимізація проекту «Розробка та налагодження виробництва нової моделі смартфону на підприємстві» засобами MS Project. // Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" - Вінниця: ДонНУ імені Василя Стуса, с. 21-23. URL: https://jait.donnu.edu.ua/article/view/12242*

4. *Ishan Gaba. Here is what You need to know about Maven [Електронне видання]. URL: https://www.simplilearn.com/tutorials/maven-tutorial/what-is-maven (дата звернення: 03.12.2022).*
5. *Tutorials Point. Maven overview [Електронне видання]. URL: https://www.tutorialspoint.com/maven/maven_overview.htm (дата звернення: 03.12.2022).*
6. *Maven Build Lifecycle Explained [Електронне видання]. URL: https://medium.com/javarevisited/maven-build-lifecycle-explained-ede8494a3d48 (дата звернення: 03.12.2022).*