* Statik kod analizi: SonarQube

SonarQube yazılan kodların statik olarak analizini (bugların bulunması, code smell adı verilen genellikle source kodun karakteristiğini inceleyip, herhangi bir bir probleme yol açıp açmayacağının kontrol edilmesi, ve ayrıca güvenlik açıklarının da kontrol edilmesi) yapan bir tool dur. SonarQube çalıştırmak için Java (Oracle JRE 8 or OpenJDK 8) yüklenmesi gerekir.

Bu SonarQube tool unu kullanmak için sonar-scanner denilen bir eklenti daha kurulur ve daha sonra konfigürasyon (sonar-scanner.properties) ayarları yapılır. Default olarak 9000 portundan dış dünyaya açılır. Kurulumları gerçekleştirdikten sonra yeni bir proje oluşturulur ve daha sonrasında sonar-scanner dosyası çalıştırılarak bir analyze report olusturulur.Bu raporda da projede bulunan bug, code smell hakkında bilgi verilir ve genel olarak quality hesaplanır. Ayrıntılı olarak bug ları vs. incelemek için her birine ayrıntılı görerek sorunlar çözülür ve böylece development süreçlerinde kodun kalitesini kontrol ederek daha iyi bir proje elde edilir. [1]

* Kubernetes yapısı

Kubernetes containerized çalışmaları, servisleri ve bunların bütün konfigürasyon ve otomasyonunu yöneten bir open source platformdur. Kubernetes in sağladığı faydalar şu şekilde sıralanabilir :

i. Dağıtık sistemleri esnek bir biçimde çalıştırma için bir framework sağlama

ii. Aplikasyonlara ait failover, deployment gibi işlemler

iii. Load Balancing

iv. Automated rollouts and rollbacks

v. Automated bin packing

vi. Self-healing

vii. Secret and configuration management

Kubernetes i deploy edebilmek için bir cluster yapısının olması gerekir, ve bu her bir cluster da worker node adı verilen bir yapı vardır.

Bu worker node lar aplikasyonun çalışmasını sağlayan Pods adı verilen yapıyı barındırır.

Kubernetes de bulunan control plane yapısı genel olarak container lara ait tanımlama, deploy etme, yönetme gibi işlevleri vardır. Daha önceki teknolojilerde düşünürsek master server olarak tanımlanabilir. Bu control plane cluster da olan bütün worker node ları ve Pod ları yönetir.

Prod ortamında genellikle birçok makine üzerinde çalışır, böylece high-availability ve fault-tolerance sağlar[2].

* Orchestration yapısı

Container Orchestration olarak da bilinen bu yapı genel olarak containerized uygulamalarının otomasyonunu, deployment ini ve yönetilmesini sağlayan bir yapıdır.

Bu yapının sürdürülmesi için birçok farklı metodoloji ve tool vardır fakat genel olarak uygulaması 3 adımda gerçekleştirilir.

-İlk olarak developer bir konfigürasyon dosyası yazar (JSON or YAML) ve bu dosyada uygulamalar için hangi container image larının kullanıldığı ve nerede saklandığı belirtilir.Network bağlantıları ayarlanır.Versiyonlama belirtilir.

-Daha sonra orchestration tool bu konfigürasyon dosyasını alarak host üzerinde CPU, capacity, memory ve diğer constraintleri de hesaplayarak en uygun host u seçerek deployment i gerçekleştirir.

- Deployment gerçekleştirildikten sonra, orchestration tool life-cycle yönetimini de gerçekleştirir, ve bu süreçte scalability(up and down), load balancing i yönetir ve uygulamanın erişilebilir olduğundan emin olup performans analizi yapar, ve logları toplayarak, uygulamayı monitor eder, ve eğer herhangi bir performance issue varsa bunu bildirir. Ve ya biz de kendimiz plugin monitor servisleri ekleyerek de sistemi daha iyi takip edebiliriz. [3]

En popüler olan container orchestration tool u Kubernetes dir. Bunun yanı sıra Docker Swarm, Apache Mesos gibi tool lar da vardır. [4]

* Npm nedir ve niçin kullanılır?

Node Package Manager ya da Node Packaged Modules olarak da bilinen javascript için geliştirilen bir paket yöneticisidir [5].

Npm genel olarak third-party packetlerin yüklenmesini, sistemdeki paketlerin de yönetilmesini(delete, list, update) sağlayan ve komut satırı üzerinden çalışan bir uygulamadır.

Npm i kurabilmek için öncelikle nodejs kurulması gerekiyor, nodejs ile birlikte geldiği için nodejs yükledikten sonra kontrolü yapılabilir.

Npm le birlikte paket yüklemek için ilk olarak package.json isminde proje hakkında bilgiler girilen bir dosya oluşturmak gerekiyor. Daha sonrasında local veya global olarak paket yüklenebilir.

Diğer package manager lara örnek olarak yarn ve bower söylenebilir[6].

* Agile-Scrum nedir?

Agile modeli yazılım geliştirme süreçlerinin iyileştirilmesi üzerine kurulan bir modeldir. 2001 yılında 17 bağımsız yazılım geliştiriciler, farklı disiplinleri de ele alarak ortak bir kültür de oluşturarak herkesin kullanabileceği bir manifesto yayınladı. Agile Manifesto olarak bilinen bu manifestonun 4 temel ilkesi vardır[7]:

* Süreçler ve araçlardan ziyade bireyler ve etkileşimlere
* Kapsamlı dökümantasyondan ziyade çalışan yazılıma
* Sözleşme pazarlıklarından ziyade müşteri ile işbirliğine
* Bir plana bağlı kalmaktan ziyade değişime karşılık vermeye değer vermeye kanaat getirdik.

Agile Manifesto’ sunda belirtilen ilkeler aslında daha önce de vardı ve yazılım geliştirme süreçlerinde bunlar kullanılıyordu ve daha sonra manifesto yayınladı ve insanlar yeni bir yazılım geliştirme sürecinde bu ilkeleri uygulayarak kendi kültürlerine uyguladılar[8].

Scrum ise Agile proje yönetiminde kullanılan metodolojilerinden biridir. Agile ve scrum genellikle aynı şey olarak bilinir fakat scrum agile in framework u olarak düşünülebilir. Şöyle ki scrum framework ü sürekli öğrenmeyi ve değişken faktörlere uyumu baz alır. Böylece değişen durumlara ve gereklilikler konusunda takıma yardımcı olan ve kısa dönemli öğrenme ve geliştirme imkanı sunan bir yapıdır. Bu bağlamda **sprint** adı verilen haftalık çalışma planlarıyla hızla değişen ihtiyaçlara cevap verebilmektedir. Her sprint te yapılacak işler, takım içindeki insanlar tarafından belirlenir ve sprint sonunda bir ürün haline düşünür.

Scrum eventleri temel olarak şu şekilde sıralanabilir:

i. Organize the backlog: Genellikle product owner in market ve müşteri tarafından alınan feedback leri development team e iletmesi ve böylece daha önceki yapılmış işlerin de takibinin sağlandığı bir yapı olarak tanımlanabilir.

ii. Sprint planning: Yapılacak olan sprint in planlanması olarak açıklanabilir ve çoğunlukla scrum master in öncülüğünde gereksinimler, ihtiyaçlar, hedefler belirlenerek bir plan ortaya konur.

iii. Sprint : Bir geliştirme için planlanan zaman dilimi olarak düşünülebilir ve genellikle 1-2 hafta sürer.Daha komplex yazılım geliştirme süreçlerinde bu zaman diliminin daha kısa olması tavsiye edilir.

iv. Daily scrum or stand up : Daily görüşmeler olarak tanımlanabilir ve genellikle dün neler yapıldığı, bugün ne yapılacağı gibi sorulara cevap verilerek günlük planlama yapılır.

v. Sprint review: Sprint in sonunda, bilgi amaçlı toplantı gerçekleştirilir ve bu toplantıda yapılan işlerin bir özetini demo olarak gösterilir ve bir sonraki sprint in planlanması yapılır.

vi. Sprint retrospective: Bu session da takım bir araya gelerek işle ilgili feedbackler verilir, ve sprint boyunca nelerin işe yaradığı ve nelerin geliştirilmesine dair bir görüşme gerçekleştirilir[9].

Referanslar:

[1]https://medium.com/code-to-express/sonarqube-static-code-analysis-8b6ca5226d2b#:~:text=SonarQube%20is%20an%20open%20source,C%2B%2B%2C%20COBOL%20and%20more.

[2] <https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/>

[3] <https://www.ibm.com/cloud/learn/container-orchestration>

[4] <https://www.redhat.com/en/topics/containers/what-is-container-orchestration>

[5] <https://www.npmjs.com/about>

[6] <https://medium.com/@busramemis/npm-nedir-bc08c9610df2>

[7] <https://agilemanifesto.org/iso/tr/manifesto.html>

[8]https://medium.com/@PeopleBox/agile-nedir-scrum-nedir-ba%C5%9Far%C4%B1l%C4%B1-proje-y%C3%B6netimi-y%C3%B6ntemleri-nelerdir-64c4ae723496

[9] <https://www.atlassian.com/agile/scrum>