YAZILIM YAŞAM DÖNGÜSÜ

Nedir

Bir yazılım projesinin hayat döngüsünü belirleyen bir dizi aşamayı tanımlayan ve yöneten sistemlerdir. Yazılım yaşam döngüsü modelleri, yazılım geliştirme sürecinin tüm aşamalarını içerir ve bu aşamaların belirlenmesi, tasarlanması, uygulanması, test edilmesi, dağıtılması ve sürdürülmesini kapsar.

Aşamaları

Yazılım geliştirme yaşam döngüsü genel olarak 5 aşamadan oluşmaktadır. Bunlar; planlama, analiz, tasarım, test ve bakım alanlarından oluşmaktadır

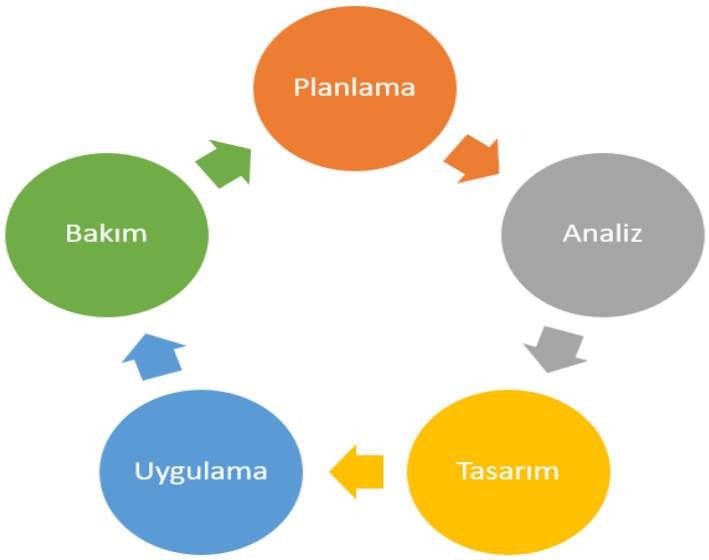
Planlama: Yazılım yaşam döngüsünün ilk aşamasıdır. Temel ihtiyaçlar belirlenir, proje için fizibilite çalışmaları yapılır (maliyetlerin ve sistemin yararlarının tanımlanması) ve proje planlaması gerçekleştirilir.

Analiz: Genelde istenilen fikrin ne olduğu ve temel analiz üzerinde konuşulacak kavramların tanımlandığı aşamadır. Çünkü aynı fikirden bahsedilmiyorsa farklı sonuçlara varılabilir. Bu analiz üzerine bir tasarım vardır. Bu analiz aşaması olarak da düşünülebilir. Problemin tanımlandığı veya yaşam döngüsünün tanımlandığı sistemin veyahut yazılımın tanımlandığı aşamadır. Analiz aşamasında projede nelerin istenildiği ile ilgili analiz çalışmaları da yapılabilir.

Tasarım: Analiz kısmının tamamlanması sonucunda tasarım aşamasına geçilir. Burada "İstediğimizi nasıl elde edeceğiz" sorusuna cevap aranır. Proje sürecinin nasıl devam edeceği konusunda bir tasarlama işlemi gerçekleşir. Tasarımda en önemli tekniklerden bir tanesi de Soyutlama(Abstraction) tekniğidir. Bu teknik ile problemi daha basit hale getirerek problemdeki önemli kısımlara dikkat edilmesine olanak tanır.

Test: Yazılım geliştirilmesi tamamlandıktan sonra müşteriye sunmadan önce, test ekibi tarafından beta testlerinin gerçekleştirilmesi aşamasıdır. Bu aşama tamamlandıktan ve varsa hatalar giderildikten sonra proje yayına alınmaktadır.

Bakım: Proje yayına alındıktan sonra oluşabilecek hataların giderilmesi, yazılımın iyileştirilmesi ve yeni işlevlerin eklenmesi süreçleridir. Bu süreç zarfında kullanıcılardan gelen bilgiler doğrultusunda bu istekler gerçekleştirilmektedir.



1.

En Çok Tercih Edilen Modeller

Şelale Modeli:

Yazılım geliştirme sürecinin belirli aşamalarının ardı ardına gerçekleştirildiği bir geliştirme modelidir. Bu model, bir sonraki aşamanın önceki aşamanın tamamlanmasından sonra başladığı bir lineer (doğrusal) süreçtir. Şelale Modeli, geleneksel bir yazılım geliştirme modelidir. Bu model, projenin gereksinimleri, tasarımı, geliştirilmesi, test edilmesi ve dağıtımı gibi aşamaların sıralı olarak gerçekleştirildiği bir süreçtir. Bu modelde, bir sonraki aşama, önceki aşamanın tamamlanmasından sonra başlar ve geri dönüş yapılamaz.

Avantajları

kullanımı ve yönetimi kolaydır
İyi belgelere sahip bir yaklaşım gerektirir
Düşük maliyetlidir.
Fazların net bir şekilde sınırlandırılması

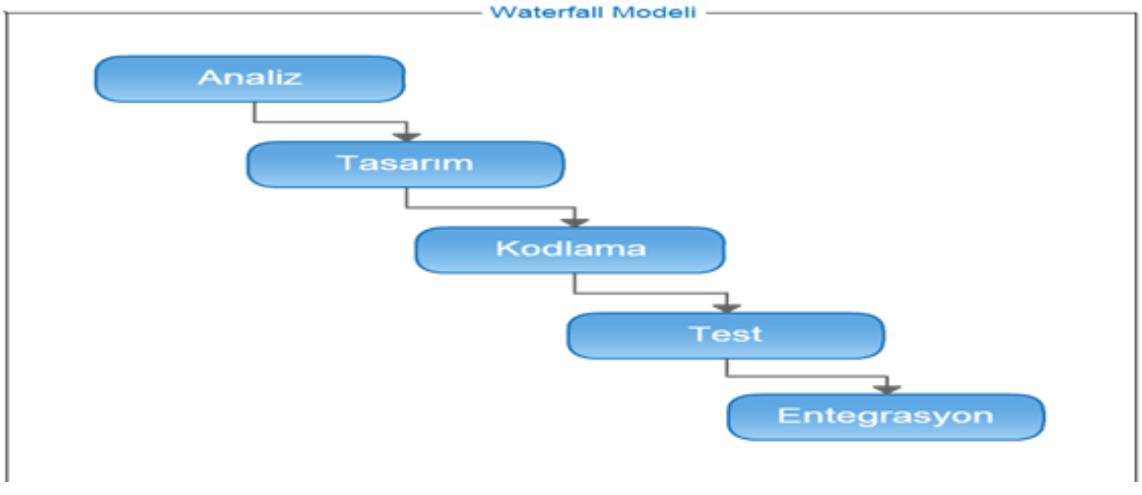
Dezavantajları

Kullanıcı katılımı sadece tanım aşamasınc
#Değişiklikler kolayca yerine getirilemez.
Beklenmedik riskleri kolayca ele almaz
Projenin bitimine kadar çalışan ürün yoktur.
Müşteri memnuniyetini sağlamak çok zordur çünkü gelişim ve değişime açık bir model değildir

Hangi Projelerde Kullanılır

Şelale modeli özellikle karmaşık projelerde kullanılır. Proje süreci boyunca gereksinimlerin çok fazla değişmediği veya değişiminin çok zor olduğu projelerde tercih edilir. Örneğin, savunma, havacılık, tıp, endüstriyel üretim gibi sektörlerde şelale Model sıklıkla kullanılmaktadır.

Diğer Modellerden Farkı



V Modeli:

Şelale modelinin bir türevi olarak düşünülebilir. Aynı aşamalar sırayla gerçekleştirilir. Ancak her aşamadan sonra bir test aşaması yer alır. Doğrusal bir yönde ilerlemek yerine, süreç adımları kodlama evresinden sonra yukarıya doğru eğim alır ve tipik V şeklini oluşturur. V-Modelinin en temel özelliği "ürünün test edilmesi kendisine karşılık gelen geliştirme aşamasına paralel olarak planlanmaktadır."

Avantajları:

#Basit ve kullanımı kolaydır
#Doğrulama ve onaylama planları erken aşamalarda vurgulanır
#Hataların bulunması erken aşamada olur
#Hataların bir sonraki aşamaya geçmesi önlenir

Dezavantajları:

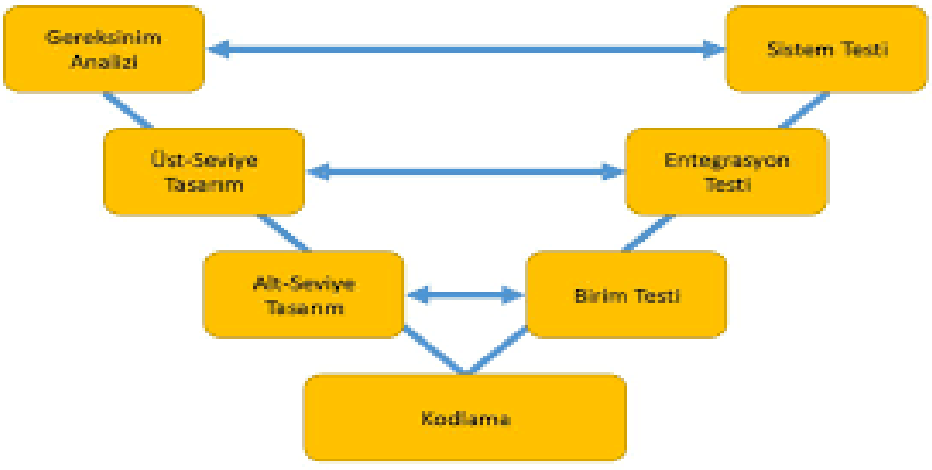
#Uygulama şekli oldukça katı, kesin kurallara bağlıdır.
Yazılım şelalede olduğu gibi geliştirme aşamasında geliştirilir, bu nedenle yazılımın erken prototipleri üretilmez.
Herhangi bir aşamada gereksinimler üzerinde değişiklik olursa, test belgelerinin de diğer belgeler ile birlikte güncellenmelidir.
Fazlar arasında tekrarlamaları kullanmaz
Geliştirme devam ettikçe iş ve ürün gereksinimleri de değişkenlik gösterebilir

Hangi Projelerde Kullanılır

Bu model, genellikle büyük ölçekli, karmaşık ve kritik yazılım projelerinde kullanılır Özellikle, güvenilirlik, kalite ve test süreçlerinin önemli olduğu projelerde kullanılır. Örneğin, tıbbi cihazlar, savunma sistemleri ve finansal yazılımlar gibi kritik önemdeki projelerde V Modeli kullanılabilir.

Diğer Modellerden Farkı

Diğer modellerden farklı olarak, test sürecini ve testleri diğer geliştirme aşamalarının bir parçası olarak ele alır ve testlerin doğru ve eksiksiz bir şekilde gerçekleştirilmesine özel bir önem verir.



Spiral Model

Yinelemeli geliştirme süreci modelini Şelale modelinin öğeleriyle birleştiren, risk yönetimi için kullanılan bir modeldir. Bir döngüsel yaklaşım benimser ve her döngüde yazılımın bir önceki versiyonuna eklenen yeni işlevselliği içerir. Her döngü, önceki döngüden elde edilen geri bildirimlere dayanarak düzenlenir. Gelişim sürecinde sürekli işlevsellik eklendiği için spiraldir.Her aşama bir tasarım hedefi ile başlayıp müşteri incelemesi ile biter

Avantajları:

Sürekli geliştirme sağlandığı için risk yönetimi kolaylaşır.
Geliştirme hızlıdır ve özellikler sistematik bir şekilde eklenir.
Daha büyük projeler / yazılımlar stratejik bir şekilde oluşturulur ve yönetilir
Yazılım erken üretilir.
Geliştirme hızlıdır

Dezavantajları:

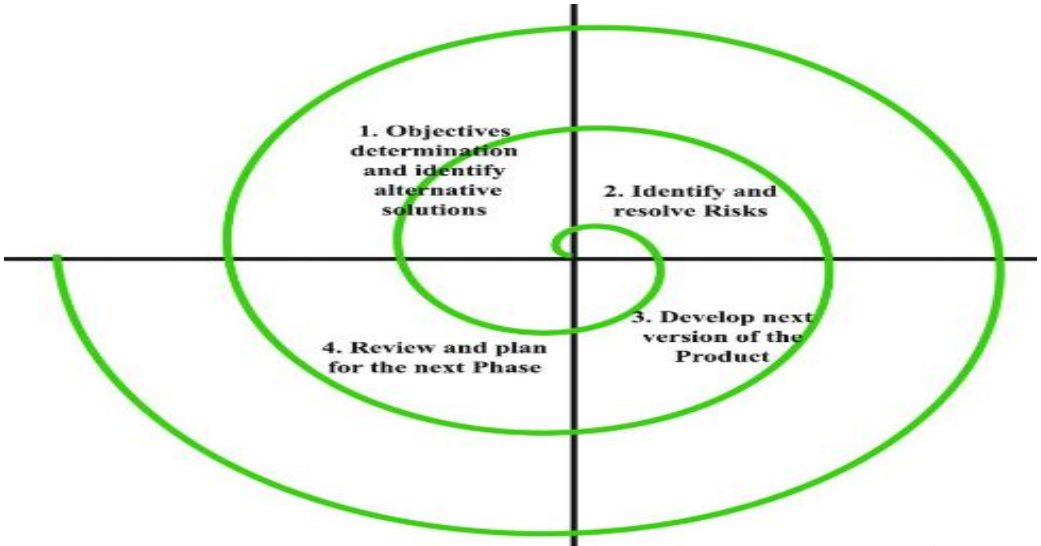
Spiral sonsuza kadar gidebilir.
Küçük projeler için maliyetlidir.
Düzgün çalışması için spiral model protokolüne kesinlikle uyulmalıdır.
Ara aşamalara sahip olduğu için dokümantasyon daha fazladır.
Risk analizi önemli bir aşamadır, bu nedenle uzman kişiler gerektirir.

Hangi Projelerde Kullanılır

Spiral model, büyük ve karmaşık yazılım projelerinde kullanılmak için tasarlanmıştır. Özellikle, belirsiz gereksinimleri olan projeler veya yeni teknolojilerin kullanıldığı projeler gibi riskleri yüksek olan projelerde spiral model tercih edilebilir. Ayrıca, projenin yaşam döngüsü boyunca önemli değişikliklerin beklenildiği projelerde de spiral model kullanılabilir. Bunun nedeni, spiral modelin esnek olması ve her döngüde yapılan değişiklikleri yönetebilmesidir. Ancak, küçük ölçekli projelerde veya düşük riskli projelerde spiral model kullanmak gereksiz olabilir.

Diğer Modellerden Farkı

Spiral model, diğer yazılım geliştirme modellerinden farklıdır çünkü bir döngüsel ve tekrarlayan bir yaklaşımı benimser. Spiral model, diğer modellere göre daha esnek bir yaklaşım sunar ve her döngüde geliştirme sürecinde geriye dönük bir hareket yapılmasına izin verir. Ayrıca, risk yönetimi önemli bir bileşen olarak ele alınır ve her döngüde projenin riskleri ve zorlukları incelenir. Bu, spiral modelin, diğer modellere göre daha riskli projelerin geliştirilmesinde tercih edilmesinin nedenlerinden biridir.



Kodla ve Düzelt Modeli

Çoğunlukla öğrenci projelerinde kullanılan herhangi yöntem içermeyen, dökümantasyon gerektirmeyen proje türüdür. Çoğunlukla sonuç olarak spagetti kod elde edilir.
Genellikle resmi olmayan bir ürün fikriyle başlar ve program ürün hazır olana kadar ya da gerekli zaman bitene kadar kodlama yapılarak devam edilir.

Avantajları:

Herhangi bir planlamaya ihtiyaç duyulmaz
Uzman görüşüne ihtiyaç düşüktür, herkes bu modeli kullanabilir.
hatanın tespiti ve düzeltilmesi konusunda yüksek doğruluk sağlar
veri iletimi veya depolama sırasında oluşabilecek hataları tespit ederek ve düzelterek veri kaybını önler.
Yazılım geliştirmenin en kolay yoludur
#Daha az planlama gerektirir:

Dezavantajları:

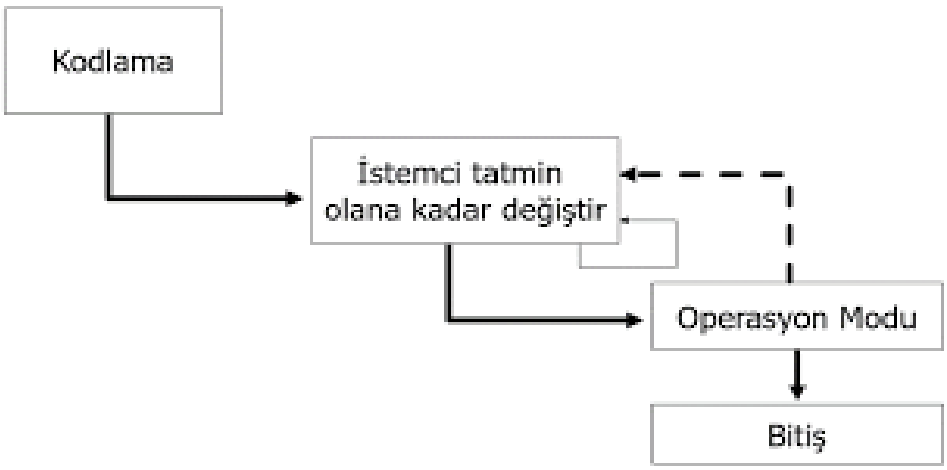
Bitiş süresi belli değildir.
Kodları düzeltmek maliyetli olabilir.
#Daha fazla zaman ve kaynak gerektirir
#Hataların tamamen çözülmesini garanti etmez
#Kodda yaygın hataların oluşmasına neden olabilir

Hangi Projelerde Kullanılır

"Kodla ve düzelt" yaklaşımı, hızlı teslimat gerektiren, küçük veya orta ölçekli projeler için uygundur. Bu yaklaşım, hataların tespit edilmesi ve giderilmesine odaklanır ve sürekli test ve düzeltme süreci içerir. Ancak daha büyük ve karmaşık projeler için uygun değildir ve zaman veya bütçe kısıtlamaları olan projelerde de kullanılması zor olabilir.

Diğer Modellerden Farkı

Yazılım geliştirme modelleri, genellikle şu adımlardan oluşur: planlama, tasarım, geliştirme, test etme ve dağıtım. Bu modellerde, her adımın ayrı ayrı tamamlanması gerektiği ve bir sonraki adıma geçilmeden önce önceki adımların tamamlanmış olması gerektiği kabul edilir. "Kodla ve düzelt" yaklaşımı ise, hataların tespit edilmesi ve giderilmesine odaklandığı için, diğer adımlara daha az önem verir. Bu yaklaşımda, yazılım sürekli olarak test edilir ve gerekli düzeltmeler yapılır. Bu nedenle, planlama ve tasarım aşamalarına daha az zaman ayrılır ve yazılım geliştirme süreci daha esnek hale gelir



Scrum Modeli

Scrum, yazılım geliştirme sürecinde kullanılan bir çerçevedir ve Agile Manifesto'nun prensiplerine dayanır. Scrum, özellikle karmaşık ve değişken projelerde kullanılan bir yaklaşımdır ve proje ekibinin sürekli olarak müşteri geri bildirimlerine dayalı olarak çalışmasına ve hızlı bir şekilde sonuçlar üretmesine olanak tanır. Scrum modeli, üç ana bileşenden oluşur: ürün sahibi, scrum ustası ve geliştirme ekibi. Ürün sahibi, proje için gereksinimleri belirler ve müşteri geri bildirimlerini yönetir. Scrum ustası, takımın Scrum prensiplerine uygun olarak çalışmasını sağlar ve geliştirme ekibi, ürün sahibinin belirlediği gereksinimleri karşılamak için ürün geliştirir. Scrum modelinde, proje kısa zaman dilimlerinde, genellikle iki haftalık sürelerde iterasyonlar halinde ilerler. Her iterasyon, planlama, geliştirme, test etme ve sunum aşamalarından oluşur. Bu süreç, müşteri geri bildirimlerine dayalı olarak tekrarlanır ve ürün sahibi, müşteri gereksinimlerine uygun olarak ürünün geliştirilmesini yönlendirir. Scrum, müşteri memnuniyetini arttırmak, hızlı teslimat yapmak, değişen gereksinimlere uyum sağlamak ve proje kalitesini arttırmak gibi avantajlar sağlar. Ancak, Scrum modeli uygulanırken, takım üyelerinin disiplinli olması ve birbirleriyle iyi bir şekilde iletişim kurması önemlidir. Ayrıca, Scrum modelinin uygulanması ve sürdürülmesi, bir Scrum ustasının liderliğinde ve proje ekibinin sürekli olarak gelişmesi ile mümkündür.

Avantajları:

müşteri memnuniyetini artırır
hızlı teslimat sağlar
değişen gereksinimlere uyum sağlar
proje kalitesini arttırır.

Dezavantajları:

yetersiz eğitim ve deneyimle etkisiz olması
bazı projeler için uygun olmaması
belirsizlik olmaması durumunda verimliliği azaltması
akım üyeleri arasında koordinasyon ve iletişim eksikliği yaşanması

Hangi Projelerde Kullanılır

Scrum, özellikle karmaşık ve değişken projelerde başarılı olabilen bir proje yönetim metodolojisidir. Scrum modeli, iterasyonel ve inkremental bir yaklaşım benimser ve müşteri geri bildirimlerine dayalı çalışmanın önemli olduğu projelerde etkilidir. Bu nedenle, ürün gereksinimlerinin sık sık değişebileceği ve önceliklerin belirsiz olabileceği projelerde kullanılması uygundur. Scrum modeli ayrıca, esneklik gerektiren projelerde de tercih edilebilir. Scrum, proje sürecindeki değişiklikleri yönetmek için bir çerçeve sunar ve takımın esneklikle yanıt vermesine olanak tanır. Bu özellikleri sayesinde, Scrum modeli genellikle yazılım geliştirme, ürün tasarımı ve inovasyon projeleri gibi dinamik ve hızlı bir ortam gerektiren projelerde kullanılır.

Diğer Modellerden Farkı

Scrum modeli, diğer proje yönetimi modellerinden farklı olarak, müşteri odaklı bir yaklaşım benimser ve iterasyonlar halinde çalışır. Bu model, ürün sahibi, geliştirme ekibi ve Scrum Master'dan oluşan bir takımı kullanarak, müşteri ihtiyaçlarını anlamaya ve geliştirme sürecini optimize etmeye odaklanır. Scrum modeli, diğer modellerden farklı olarak, iş gereksinimlerinin sürekli olarak değişebileceği dinamik projelerde etkili olabilen bir esneklik sağlar. Ayrıca, Scrum modeli, geliştirme ekibine özerklik vererek, karar verme sürecine dahil etmeye odaklanır ve takımın kendi kendini yönetmesine izin verir.Diğer modellerde ise, planlama ve uygulama ayrı ayrı aşamalarda gerçekleştirilir ve değişiklikler sınırlıdır. Bu nedenle, Scrum modeli diğer modellere göre daha hızlı, daha esnek ve müşteri odaklı bir yaklaşım benimser.

Günümüzde Neden Popüler

scrum modelinin günümüzde daha popüler olmasının birkaç nedeni vardır.İlk olarak, Scrum modeli, müşteri ihtiyaçlarını sürekli olarak göz önünde bulundurarak, müşteri odaklı bir yaklaşım benimser ve müşteri memnuniyetine büyük önem verir. Bu nedenle, müşteriye öncelik veren ve onların değişen ihtiyaçlarına hızla adapte olabilen Scrum modeli, özellikle yazılım geliştirme gibi hızlı değişen ve dinamik projelerde popülerdir. İkinci olarak, Scrum modeli, geliştirme ekibine özerklik verir ve karar verme sürecine dahil eder. Bu sayede, takım motivasyonu artar ve daha yaratıcı ve yenilikçi çözümler üretmeleri sağlanır. Bu da projelerin daha başarılı olmasını sağlar. Üçüncü olarak, Scrum modeli, iş gereksinimlerinin sürekli olarak değişebileceği dinamik projelerde etkili olabilen bir esneklik sağlar. Bu sayede, proje yöneticileri ve geliştiriciler, hızlı değişen piyasa koşullarına uyum sağlamak ve hızla değişen müşteri ihtiyaçlarına yanıt vermek için esnek bir yaklaşım benimseyebilirler. Son olarak, Scrum modeli, sıklıkla yazılım geliştirme, ürün tasarımı ve inovasyon projelerinde kullanılmaktadır. Bu nedenle, bu alanlarda çalışan profesyoneller arasında popülerdir ve bu alanlarda çalışan şirketlerin tercih ettiği bir proje yönetim metodolojisi haline gelmiştir.

modellerin karşılaştırılması

no	Model/özellik	Şelale model	V modeli	Spiral Model	Kodla ve düzelt Mode	Scrum modeli
1	**Maliyet**	Maliyetli	Maliyetli	Maliyetli	Düşük	Çok yüksek
2	**Basitlik**	Basit	Orta	Karmaşık	Basit	Karmaşık
3	**Risk Duyarlılığı**	Yüksek	Yüksek Değil	Düşük	Yüksek	Azaltılmış
4	**Başarı Garantisi**	Düşük	Orta	Yüksek	Düşük	Çok yüksek
5	**Değişiklik Yapma**	Zor	Zor	Kolay	Kolay	Zor
6	**Gereksinin Belirleme**	Başlangıç	Başlangıç	Belirli Sıklıkla	Başlangıç	Belirli sıklıkla
7	**Zaman**	Çok uzun	Uzun	Uzun	Çok uzun	Kısa

KAYNAKÇA

1) https://medium.com/@denizkilinc/yaz%C4%B1l%C4%B1m-ya%C5%9Fam-d%C3%B6ng%C3%BCs%C3%BC-temel-a%C5%9Famalar%C4%B1-software-development-life-cycle-core-processes-197a4b503696

2)https://medium.com/@brfn.kcr26/yazilim-geli%CC%87%C5%9Fti%CC%87rme-ve-s%C3%BCre%C3%A7-modelleri%CC%87-2131ea5f09b2#:~:text=1.Kodla%20Ve%20D%C3%BCzelt%20Ya%C5%9Fam,kadar%20kodlama%20yap%C4%B1larak%20devam%20edilir.&text=%C3%BC%20%C3%87ok%20k%C3%BC%C3%A7%C3%BCk%20projelerde%20ya,%C3%BC%20Program%20a%C5%9Famalar%C4%B1%20%C3%A7abuk%20ge%C3%A7ilir.

3) https://talentgrid.io/tr/yazilim-gelistirme-modelleri/