A.PROJENİN TANIMI:  
Bu proje, Türkçe metinlerin analiz edilip sınıflandırılması, veritabanına kaydedilmesi ve kullanıcılara ilgili bilgiler sunulmasını amaçlayan bir yapay zeka uygulamasıdır. Kullanıcı, programa Türkçe metinler girerek başlar. Girilen metinler, doğal dil işleme (NLP) teknikleri kullanılarak analiz edilir ve metnin genel konusu ile alt konuları belirlenir. Bu sınıflandırma işleminin ardından, belirlenen konular ve alt konular veritabanına kaydedilir ve sonuçlar konsolda görüntülenir. Kullanıcı, programı kapatmadığı sürece yeni metinler girmeye devam edebilir ve her yeni metin için aynı süreç tekrar edilir. Program, kullanıcının girdiği tüm metinleri değerlendirerek sohbetin genel konusunu belirler. Belirlenen bu genel konu ve alt konular doğrultusunda, internet üzerinde arama motorları kullanılarak bilgi araştırması yapılır ve ilgili bilgiler kullanıcıya sunulur. Sunulan bu bilgiler de veritabanına kaydedilir. Böylece, kullanıcıya sürekli olarak güncel ve ilgili bilgiler sağlanır ve bu bilgiler sistemde saklanır. Bu proje, metin sınıflandırma, veri yönetimi ve bilgi sunumu alanlarında kapsamlı bir çözüm sunmayı hedeflemektedir.

B.PROJENİN SAHİP OLDUĞU NİTELİKLER

1. **Türkçe Metin Girişi**:
   * Kullanıcı, programa Türkçe metinler girebilir.
2. **Metin Analizi ve Sınıflandırma**:
   * Girilen metinler, doğal dil işleme (NLP) teknikleri kullanılarak analiz edilir.
   * Metinlerin genel konusu ve alt konuları belirlenir ve sınıflandırılır.
3. **Veritabanına Kayıt**:
   * Belirlenen konular ve alt konular veritabanına kaydedilir.
   * Sınıflandırma sonuçları konsolda görüntülenir.
4. **Sürekli Kullanıcı Girişi**:
   * Kullanıcı, programı kapatmadığı sürece sürekli olarak yeni metinler girebilir.
   * Her yeni metin için sınıflandırma ve kayıt işlemleri tekrar edilir.
5. **Sohbetin Genel Konusunun Belirlenmesi**:
   * Kullanıcının girdiği tüm metinler değerlendirilerek sohbetin genel konusu belirlenir.
6. **Bilgi Araştırma ve Sunma**:
   * Belirlenen genel konu ve alt konular doğrultusunda, internet üzerinde arama motorları kullanılarak bilgi araştırması yapılır.
   * Araştırılan bilgiler kullanıcıya sunulur.
7. **Sunulan Bilgilerin Kaydedilmesi**:
   * Kullanıcıya sunulan bilgiler de veritabanına kaydedilir.
8. **Kapsamlı Veri Yönetimi**:
   * Girilen metinler ve araştırılan bilgiler sistemde organize bir şekilde saklanır ve yönetilir.
9. **Kullanıcı Dostu Arayüz**:
   * Kullanıcının metin girişi yapabileceği ve sonuçları görüntüleyebileceği kullanıcı dostu bir arayüz sağlanır.
10. **Gerçek Zamanlı İşleme**:
    * Kullanıcının metin girişi yaptığı anda gerçek zamanlı olarak metin analiz ve sınıflandırma işlemleri gerçekleştirilir.
11. **Esneklik ve Genişletilebilirlik**:
    * Sistem, farklı konular ve alt konular için esnek bir yapıya sahiptir ve gelecekte genişletilebilir.

Bu nitelikler, projenin işlevselliğini ve kullanıcıya sağladığı değeri artırarak kapsamlı bir yapay zeka uygulaması olmasını sağlar.

Projenizin doğası gereği, özellikle kullanıcıdan sürekli geri bildirim almayı, metin analizlerinin doğruluğunu artırmayı ve veri tabanına sürekli olarak yeni veriler eklemeyi hedeflediğiniz için **Çevik (Agile) Yazılım Geliştirme Yaşam Döngüsü** bu proje için en uygun yaklaşımdır. İşte nedenleri:

**Çevik (Agile) Yazılım Geliştirme Yaşam Döngüsü**

**1. Esneklik ve Uyarlanabilirlik**

* **Çevik metodolojisi**, proje gereksinimlerinin ve önceliklerinin zamanla değişebileceğini kabul eder. Bu, kullanıcıdan gelen geri bildirimlere göre sistemde hızlı değişiklikler yapmanıza olanak tanır.
* Projenizin metin analizi ve sınıflandırma gibi karmaşık NLP işlemleri içerdiğini göz önüne alırsak, algoritmaların performansını sürekli iyileştirmek için esnek bir süreç gereklidir.

**2. Sürekli Geri Bildirim**

* Kullanıcıdan sürekli olarak metin girdisi alacağınız ve bu girdilere dayanarak sistemin performansını değerlendireceğiniz için, **çevik metodolojisi** ile kısa geri bildirim döngüleri oluşturabilirsiniz.
* Her iterasyonda (sprint), kullanıcı geri bildirimlerine dayanarak iyileştirmeler yapabilir ve yeni özellikler ekleyebilirsiniz.

**3. İteratif ve Artırımlı Geliştirme**

* Proje, belirli bir sırayla tamamlanan birçok küçük, yönetilebilir parçaya bölünebilir. Bu sayede, her iterasyonda yeni özellikler eklenir ve mevcut özellikler iyileştirilir.
* Bu iteratif yaklaşım, metin analiz algoritmalarını ve veritabanı işlemlerini adım adım geliştirme olanağı sağlar.

**4. Müşteri (Kullanıcı) Katılımı**

* Çevik metodolojisi, kullanıcıların geliştirme sürecine aktif olarak katılmasını teşvik eder. Bu, kullanıcıların gerçek ihtiyaçlarını ve beklentilerini daha iyi anlamanızı sağlar.
* Kullanıcı geri bildirimleri, sınıflandırma doğruluğunu artırma ve bilgi sunumu işlevlerini optimize etme konusunda rehberlik eder.

**Çevik Süreçte Kullanılabilecek Pratikler**

1. **Scrum**
   * **Scrum** çerçevesi, projenizin yönetimini kolaylaştırabilir. Sprintler, günlük stand-up toplantıları, sprint incelemeleri ve retrospektifler, sürekli iyileştirme ve adaptasyon sağlar.
2. **Kanban**
   * **Kanban**, görselleştirilmiş iş akışı ile işlerinizi yönetmenize yardımcı olabilir. Bu, özellikle metin analizi ve bilgi araştırma süreçlerinin takip edilmesinde faydalı olabilir.
3. **Test Odaklı Geliştirme (TDD)**
   * **TDD** yaklaşımı, metin sınıflandırma algoritmalarının doğruluğunu sürekli olarak test etmeyi ve iyileştirmeyi sağlar.

**Özet**

Çevik yazılım geliştirme yaşam döngüsü, projenizin gerektirdiği esnekliği, kullanıcı geri bildirimlerine dayalı sürekli iyileştirmeyi ve iteratif geliştirmeyi en iyi şekilde destekler. Bu metodoloji, projenizin başarılı olmasını ve kullanıcı ihtiyaçlarına hızlıca yanıt vermesini sağlayacaktır.

Kullanılacak Metodoloji:

Agile metodolojisi, özellikle bu tür sürekli etkileşim ve geri bildirim gerektiren projeler için oldukça uygundur. Agile, esnek ve uyarlanabilir bir yaklaşımdır, bu da kullanıcı geri bildirimlerine hızlıca yanıt verebilmenizi sağlar. Bu proje için, Agile metodolojisinin bir çeşidi olan **Scrum** metodolojisini kullanmanızı öneririm. Scrum, projeyi küçük ve yönetilebilir parçalara (sprintler) bölerek, sürekli olarak gelişim ve iyileştirme sağlar.

**Neden Scrum Bu Proje İçin Uygundur?**

1. **Kullanıcı Geri Bildirimi**: Kullanıcının sürekli yeni metinler girdiği ve bu metinlerin anında işlenmesi gerektiği bir projede, kullanıcı geri bildirimleri hızlı bir şekilde toplanıp, sonraki sprintlerde bu geri bildirimlere göre iyileştirmeler yapılabilir.
2. **Değişen Gereksinimler**: Projede kullanıcıların farklı konularda metinler gireceği ve bu metinlere göre sürekli yeni bilgiler aranacağı düşünüldüğünde, Scrum metodolojisi değişen gereksinimlere hızlıca uyum sağlamayı mümkün kılar.
3. **Kademeli Teslimatlar**: Projenin çeşitli aşamalarını (metin analizi, veri tabanına kaydetme, genel konuyu bulma, internetten bilgi arama) ayrı ayrı sprintler olarak ele alarak, her bir aşamayı bitirip test edebilir ve kullanıcılardan geri bildirim alabilirsiniz.

**Scrum ile Proje Yönetimi**

**Scrum Roller**

1. **Scrum Master**: Scrum süreçlerini yöneten ve takımın Scrum prensiplerine uymasını sağlayan kişidir.
2. **Product Owner**: Proje gereksinimlerini belirleyen ve önceliklendiren kişidir. Bu projede kullanıcı gereksinimlerini toplar ve Backlog'u yönetir.
3. **Geliştirme Ekibi**: Yazılım geliştirme, test etme ve diğer teknik görevleri gerçekleştiren ekiptir.

**Scrum Artefaktları**

1. **Product Backlog**: Tüm yapılacak işlerin ve gereksinimlerin listelendiği yerdir. Örneğin, "metin analizi algoritması geliştirme", "veri tabanı tasarımı", "internet arama entegrasyonu" gibi maddeler bulunur.
2. **Sprint Backlog**: Her sprintte yapılacak işlerin listelendiği yerdir. Sprint süresi genellikle 2-4 hafta arasıdır.
3. **Increment**: Her sprintin sonunda tamamlanan ve potansiyel olarak kullanılabilir olan ürün parçasıdır.

**Scrum Seremonileri**

1. **Sprint Planlama**: Sprint başında yapılan ve o sprintte yapılacak işlerin belirlendiği toplantıdır.
2. **Günlük Scrum**: Takımın günlük olarak gerçekleştirdiği, 15 dakikalık kısa toplantılardır. Bu toplantılarda, dün ne yaptınız, bugün ne yapacaksınız ve engeller nelerdir soruları yanıtlanır.
3. **Sprint Review**: Sprint sonunda yapılan, sprintte tamamlanan işlerin gözden geçirildiği ve geri bildirimlerin alındığı toplantıdır.
4. **Sprint Retrospective**: Takımın süreçlerini değerlendirdiği ve iyileştirme önerileri geliştirdiği toplantıdır.

**Projenin Scrum ile Yönetimi**

1. **Proje Başlangıcı**:
   * **Product Backlog** oluşturulur ve tüm gereksinimler listelenir.
   * **Sprint Planlaması** yapılır ve ilk sprintte yapılacak işler belirlenir.
2. **İlk Sprint**:
   * Metin analizi algoritması geliştirilir ve test edilir.
   * Veritabanı tasarımı yapılır ve temel veri yapıları oluşturulur.
   * Günlük Scrum toplantıları ile ilerleme takip edilir.
   * Sprint sonunda, tamamlanan işler **Sprint Review** toplantısında gözden geçirilir ve geri bildirimler alınır.
   * **Sprint Retrospective** toplantısında süreçler değerlendirilir ve iyileştirme önerileri belirlenir.
3. **İkinci ve Sonraki Sprintler**:
   * Kullanıcı geri bildirimlerine göre yeni özellikler eklenir ve mevcut özellikler iyileştirilir.
   * İnternet arama entegrasyonu yapılır ve bilgi toplama işlevi eklenir.
   * Tüm metinlerin değerlendirilip genel konunun bulunması özelliği geliştirilir.

Yazılım Yaşam Döngüsü Nedir, Adımları:  
  
  
Yazılım yaşam döngüsü (Software Development Life Cycle, SDLC), bir yazılım ürününün geliştirilmesinden bakımına kadar olan süreçleri tanımlayan ve bu süreci adım adım yöneten bir modeldir. SDLC, yazılım geliştirme sürecini daha düzenli, verimli ve kaliteli hale getirmek amacıyla kullanılır. SDLC, yazılımın ihtiyaçlarının belirlenmesi, tasarımı, kodlanması, test edilmesi, kurulması ve bakımının yapılması gibi aşamaları içerir

**1. Planlama**

Bu aşama, yazılım projesinin temelinin atıldığı ve tüm sürecin organize edildiği aşamadır.

* **Proje Kapsamının Belirlenmesi**: Projenin amacı, hedefleri ve kapsamı netleştirilir.
* **Kaynak Planlaması**: Proje için gerekli olan kaynaklar (insan gücü, zaman, maliyet vb.) belirlenir ve planlanır.
* **Risk Analizi**: Proje süresince karşılaşılabilecek potansiyel riskler belirlenir ve bu risklere karşı önlemler alınır.
* **Zaman Çizelgesi**: Proje için detaylı bir zaman çizelgesi oluşturulur ve önemli kilometre taşları (milestones) belirlenir.

**2. Sistem Çözümleme**

Bu aşamada, yazılımın gereksinimleri detaylı bir şekilde analiz edilir ve belgelenir.

* **Gereksinim Toplama**: Kullanıcılar, paydaşlar ve diğer ilgili taraflarla görüşmeler yapılarak sistemin işlevsel ve işlevsel olmayan gereksinimleri toplanır.
* **Gereksinim Belirleme ve Dokümantasyon**: Toplanan gereksinimler netleştirilir ve detaylı bir şekilde dokümante edilir. Gereksinim belgeleri, yazılımın nasıl çalışması gerektiğini tanımlar.
* **Gereksinimlerin Onaylanması**: Belirlenen gereksinimler, tüm paydaşlarla birlikte gözden geçirilir ve onaylanır.

**3. Tasarım**

Bu aşama, yazılımın nasıl gerçekleştirileceğinin planlandığı ve tasarlandığı aşamadır.

* **Sistem Tasarımı**: Yazılımın genel mimarisi ve yapı taşları (modüller, veri tabanları, arayüzler vb.) belirlenir.
* **Detaylı Tasarım**: Her bir modülün, bileşenin ve işlevin nasıl çalışacağı detaylandırılır. Veri yapıları, algoritmalar ve kullanıcı arayüzü tasarımı bu aşamada belirlenir.
* **Prototip Geliştirme**: Gerektiğinde, kullanıcı arayüzü veya belirli işlevler için prototipler geliştirilir ve kullanıcılardan geri bildirim alınır.

**4. Gerçekleştirim (Implementation)**

Bu aşamada, tasarımda belirlenen yapıya uygun olarak yazılım kodlanır ve geliştirilir.

* **Kodlama**: Yazılımın işlevsel parçaları, belirlenen programlama dilleri ve araçları kullanılarak kodlanır.
* **Versiyon Kontrolü**: Kodlama sırasında, versiyon kontrol sistemleri (örneğin, Git) kullanılarak kod değişiklikleri takip edilir.
* **Kod İncelemeleri ve Birim Testleri**: Geliştirilen kodlar, kalite ve doğruluk açısından incelenir ve birim testleri uygulanarak işlevsellikleri doğrulanır.

**5. Yazılım Doğrulama ve Geçerleme (Verification and Validation)**

Bu aşama, yazılımın doğruluğunun ve gereksinimlere uygunluğunun test edildiği aşamadır.

* **Doğrulama (Verification)**: Yazılımın doğru bir şekilde tasarlandığını ve gereksinimlere uygun olarak geliştirildiğini doğrulamak için yapılan testlerdir. Bu testler, yazılımın iç işleyişinin doğru olup olmadığını kontrol eder.
* **Geçerleme (Validation)**: Yazılımın kullanıcı ihtiyaçlarını ve gereksinimlerini karşıladığını doğrulamak için yapılan testlerdir. Bu aşama, kullanıcı perspektifinden yazılımın doğruluğunu kontrol eder.
* **Test Türleri**: Birim testleri, entegrasyon testleri, sistem testleri ve kullanıcı kabul testleri (UAT) bu aşamada gerçekleştirilir.

**6. Kurulum ve Bakım**

Bu aşamada, yazılımın son kullanıcılar tarafından kullanılabilir hale getirilmesi ve sonrasında yapılacak bakım faaliyetleri yer alır.

* **Kurulum (Deployment)**: Yazılım, üretim ortamına kurulur ve kullanıcılar tarafından kullanılmaya başlanır. Bu süreç, yazılımın başarılı bir şekilde çalışmasını sağlamak için gerekli yapılandırmaların yapılmasını içerir.
* **Kullanıcı Eğitimi ve Destek**: Kullanıcılar, yazılımı nasıl kullanacakları konusunda eğitilir ve gerekli dokümantasyon sağlanır.
* **Bakım ve Güncellemeler**: Yazılım kullanıldıkça ortaya çıkan hatalar düzeltilir ve gerektiğinde yeni özellikler eklenir. Performans izleme, güvenlik güncellemeleri ve periyodik bakım faaliyetleri bu aşamanın bir parçasıdır.

**Proje Planlama Adımları**

**1. Proje Kapsamının Belirlenmesi**

* **Proje Adı**: Akıllı Metin Analizi ve Sınıflandırma Sistemi
* **Proje Amacı**: Kullanıcının girdiği Türkçe metinleri analiz edip, genel ve alt konuları belirleyerek sınıflandırmak; bu bilgileri veri tabanına kaydetmek ve internet üzerinden ilgili bilgiler toplayarak kullanıcıya sunmak.
* **Proje Hedefleri**:
  + Metinlerin analiz edilmesi ve sınıflandırılması.
  + Analiz sonuçlarının veri tabanına kaydedilmesi.
  + Kullanıcıya internetten ilgili bilgiler sunulması.
  + Mobil uygulama ile kullanıcı etkileşiminin sağlanması.
  + Dağıtık sistem mimarisi kullanarak internet ve bulut üzerinde çalışabilen bir çözüm geliştirilmesi.

**2. Kaynak Planlaması**

* **Proje Ekibi** (6 Personel):
  1. **Proje Yöneticisi**: Proje planlaması ve yönetiminden sorumlu.
  2. **Sistem Analisti**: Gereksinimlerin toplanması ve analiz edilmesinden sorumlu.
  3. **Yazılım Geliştirici (Backend)**: Sunucu tarafı ve veri tabanı işlemlerinden sorumlu.
  4. **Yazılım Geliştirici (Frontend)**: Web ve mobil uygulama geliştirmeden sorumlu.
  5. **Veri Bilimci**: Metin analizi ve sınıflandırma algoritmalarının geliştirilmesinden sorumlu.
  6. **Test Mühendisi**: Yazılımın test edilmesi ve kalite kontrolünden sorumlu.
* **Gerekli Araçlar ve Teknolojiler**:
  1. **Programlama Dilleri**: Python (metin analizi ve sınıflandırma), JavaScript (frontend)
  2. **Frameworks**: Flask/Django (backend), React/Angular (web frontend), React Native/Flutter (mobil)
  3. **Veri Tabanı**: PostgreSQL veya MongoDB
  4. **Bulut Hizmetleri**: AWS, Azure veya Google Cloud
  5. **Versiyon Kontrol Sistemi**: Git/GitHub
  6. **Sürekli Entegrasyon/Dağıtım (CI/CD)**: Jenkins, Travis CI

**3. Zaman Çizelgesi**

Projenin tamamlanması 4 aylık bir süreyi kapsayacak şekilde planlanmıştır. Her sprint 2 hafta sürecektir.

* **Ay 1**:
  + **Sprint 1**: Gereksinim analizi ve sistem tasarımı
  + **Sprint 2**: Metin analizi ve sınıflandırma algoritmalarının geliştirilmesi
* **Ay 2**:
  + **Sprint 3**: Backend geliştirme (API'lar ve veri tabanı entegrasyonu)
  + **Sprint 4**: Frontend geliştirme (Web uygulaması)
* **Ay 3**:
  + **Sprint 5**: Mobil uygulama geliştirme
  + **Sprint 6**: İnternet arama entegrasyonu ve veri toplama modülü
* **Ay 4**:
  + **Sprint 7**: Sistem entegrasyonu ve testler
  + **Sprint 8**: Kullanıcı kabul testleri ve son iyileştirmeler

**4. Risk Yönetimi**

* **Teknik Riskler**:
  + Metin analizinin doğru yapılamaması: Çeşitli metin setleri üzerinde önceden testler yapılacak.
  + Veri tabanı performans sorunları: Performans optimizasyonları ve indeksleme teknikleri kullanılacak.
  + Bulut entegrasyon problemleri: Bulut hizmetleri ile önceden küçük çaplı entegrasyon testleri yapılacak.
* **Proje Yönetim Riskleri**:
  + Zamanında tamamlanamama: Sprint değerlendirmeleri ile sürekli takip ve gerekli durumlarda plan revizyonu.
  + Ekip üyelerinin uyum sorunu: Düzenli toplantılar ve takım çalışması aktiviteleri.