**Proje Dokümantasyonu: 21 Günlük Alışkanlık Takip Uygulaması**

**Doküman Versiyonu:** 1.0 **Tarih:** 3 Nisan 2025 **Yazan:** Gülçin İşidoğru

**İçindekiler**

1. **Giriş**
   * 1.1. Proje Amacı
   * 1.2. Kapsam
   * 1.3. Hedef Kitle
2. **Gereksinim Analizi**
   * 2.1. İşlevsel Gereksinimler (Functional Requirements)
     + FR1: Kullanıcı Yönetimi
     + FR2: Alışkanlık Yönetimi
     + FR3: Motivasyon ve İlerleme Takibi
     + FR4: Profil ve Ayarlar
     + FR5: Bilgilendirme
   * 2.2. İşlevsel Olmayan Gereksinimler (Non-Functional Requirements)
     + NFR1: Performans
     + NFR2: Güvenlik
     + NFR3: Kullanılabilirlik (Usability)
     + NFR4: Güvenilirlik
     + NFR5: Bakım Kolaylığı
     + NFR6: Ölçeklenebilirlik
3. **Sistem Mimarisi**
4. **Teknik Tasarım Detayları**
   * 4.1. Veritabanı Yapısı (Firebase Firestore)
   * 4.2. Kimlik Doğrulama Akışı
   * 4.3. Alışkanlık Yönetimi Akışı (Örnek: Alışkanlığı Tamamlandı İşaretleme)
   * 4.4. Tema Yönetimi
5. **Test ve Doğrulama**
6. **Dağıtım (Deployment)**

**1. Giriş**

Bu doküman, "21 Günlük Alışkanlık Takip Uygulaması" projesinin genel tanımını, hedeflerini, kapsamını, gereksinimlerini, mimarisini ve teknik detaylarını içermektedir. Uygulama, kullanıcıların yeni alışkanlıklar edinmelerine ve bu alışkanlıkları 21 gün boyunca düzenli olarak takip etmelerine yardımcı olmak amacıyla tasarlanmıştır.

**1.1. Proje Amacı**

Uygulamanın temel amacı, kullanıcılara alışkanlık edinme sürecinde motive edici ve takip edilebilir bir platform sunmaktır. "Atomik Alışkanlıklar" gibi kitaplardan ilham alan 21 günlük döngü prensibiyle, kullanıcıların kararlılıklarını sürdürmeleri ve olumlu davranışları hayatlarına entegre etmeleri hedeflenmektedir.

**1.2. Kapsam**

Bu doküman, projenin ilk sürümünü kapsar. Kapsam dahilindeki temel özellikler şunlardır:

* Kullanıcı Kaydı ve Girişi (Firebase Kimlik Doğrulama ile)
* Alışkanlık Ekleme, Görüntüleme, İşaretleme ve Silme
* 21 Günlük İlerleme Takibi ve Bölüm (Chapter) Sistemi
* Günlük Motivasyon Mesajları
* Kullanıcı Profili Görüntüleme (Toplam Alışkanlık, Tamamlanan Günler, Mevcut Bölüm)
* Tema Ayarları
* Basit Navigasyon (Ana Sayfa, Profil, Hakkında, Ayarlar)

**1.3. Hedef Kitle**

* Yeni alışkanlıklar edinmek isteyen bireyler
* Mevcut alışkanlıklarını düzenli olarak takip etmek isteyenler
* Basit, kullanımı kolay ve motive edici bir alışkanlık takip aracına ihtiyaç duyan herkes

**2. Gereksinim Analizi**

Bu bölümde, uygulamanın functional (işlevsel) ve non-functional (işlevsel olmayan) gereksinimleri detaylandırılmıştır.

**2.1. İşlevsel Gereksinimler (Functional Requirements)**

Uygulamanın gerçekleştirmesi gereken temel işlevler:

* **FR1: Kullanıcı Yönetimi**
  + FR1.1: Kullanıcılar e-posta ve şifre ile yeni bir hesap oluşturabilmelidir.
  + FR1.2: Kullanıcılar e-posta ve şifre ile mevcut hesaplarına giriş yapabilmelidir.
  + FR1.3: Kullanıcılar uygulamadan güvenli bir şekilde çıkış yapabilmelidir.
  + FR1.4: Uygulama, kullanıcıların oturum durumunu (giriş yapılıp yapılmadığını) algılamalı ve buna göre ilgili sayfaya yönlendirme yapmalıdır.
* **FR2: Alışkanlık Yönetimi**
  + FR2.1: Kullanıcılar yeni bir alışkanlık başlığı, açıklaması ve emojisi ile alışkanlık ekleyebilmelidir.
  + FR2.2: Kullanıcılar mevcut alışkanlıklarını liste halinde ana sayfada görebilmelidir.
  + FR2.3: Kullanıcılar her gün için bir alışkanlığı "tamamlandı" olarak işaretleyebilmelidir.
  + FR2.4: Kullanıcılar tamamladıkları alışkanlıkların durumunu (işaretli/işaretsiz) görsel olarak görebilmelidir.
  + FR2.5: Kullanıcılar istedikleri bir alışkanlığı listeden silebilir.
  + FR2.6: Uygulama, kullanıcının 21 günlük ilerlemesini ve mevcut bölümünü takip etmelidir.
  + FR2.7: Her gün, tüm alışkanlıkların tamamlama durumu otomatik olarak sıfırlanmalıdır.
  + FR2.8: 21 günlük döngü tamamlandığında, uygulama bir sonraki bölüme otomatik olarak geçmeli ve alışkanlıkların tamamlanma durumlarını sıfırlamalıdır.
* **FR3: Motivasyon ve İlerleme Takibi**
  + FR3.1: Kullanıcılar "Günün Motivasyonunu Gör!" düğmesine tıklayarak günlük bir motivasyon sözü görüntüleyebilmelidir.
  + FR3.2: Uygulama, genel ilerlemeyi (tamamlanan gün sayısı) bir ilerleme çubuğu ile görselleştirmelidir.
  + FR3.3: Kullanıcılar mevcut bölüm başlığını görebilmelidir.
* **FR4: Profil ve Ayarlar**
  + FR4.1: Kullanıcılar profil sayfasında toplam alışkanlık sayısını, tamamlanan gün sayısını ve mevcut bölümünü görebilmelidir.
  + FR4.2: Kullanıcılar ayarlar sayfasında uygulama temasını (örneğin: açık/koyu tema) değiştirebilmelidir.
  + FR4.3: Uygulama, tema tercihini kaydedip bir sonraki ziyarette uygulamalıdır.
* **FR5: Bilgilendirme**
  + FR5.1: Kullanıcılar "Hakkında" sayfasında uygulama hakkında genel bilgi edinebilmelidir.

**2.2. İşlevsel Olmayan Gereksinimler (Non-Functional Requirements)**

Uygulamanın performansı, güvenliği ve kullanılabilirliği ile ilgili gereksinimler:

* **NFR1: Performans**
  + NFR1.1: Sayfalar, kullanıcı etkileşiminden sonra maksimum 2 saniye içinde yüklenmelidir.
  + NFR1.2: Alışkanlık ekleme, silme, işaretleme gibi işlemler 1 saniye içinde tamamlanmalıdır.
* **NFR2: Güvenlik**
  + NFR2.1: Kullanıcı kimlik doğrulama bilgileri (e-posta/şifre) Firebase tarafından güvenli bir şekilde yönetilmelidir.
  + NFR2.2: Kullanıcı verileri (alışkanlıklar, ilerleme) yalnızca ilgili kullanıcı tarafından okunabilir ve değiştirilebilir olmalıdır.
  + NFR2.3: Cross-Site Scripting (XSS) ve Cross-Site Request Forgery (CSRF) gibi temel web güvenlik açıklarına karşı önlemler alınmalıdır.
* **NFR3: Kullanılabilirlik (Usability)**
  + NFR3.1: Uygulama arayüzü sezgisel ve kolay anlaşılır olmalıdır.
  + NFR3.2: Uygulama, farklı ekran boyutlarına (masaüstü, tablet, mobil) uyumlu olmalıdır (Responsive Design).
  + NFR3.3: Kullanıcıya geri bildirim (başarı/hata mesajları) açık ve anlaşılır bir şekilde sunulmalıdır.
* **NFR4: Güvenilirlik**
  + NFR4.1: Veritabanı işlemleri (Firestore) güvenilir olmalı ve veri kaybı yaşanmamalıdır.
  + NFR4.2: Firebase servislerinde kesinti olması durumunda, uygulama kullanıcıya bilgi vermeli ve durumu yönetmelidir.
* **NFR5: Bakım Kolaylığı**
  + NFR5.1: Kod yapısı modüler ve okunabilir olmalı, gelecekteki geliştirmelere açık olmalıdır.
  + NFR5.2: Harici kütüphaneler güncel tutulmalıdır.
* **NFR6: Ölçeklenebilirlik**
  + NFR6.1: Kullanıcı sayısı arttığında Firebase'in sağladığı altyapı sayesinde sistem ölçeklenebilir olmalıdır.

**3. Sistem Mimarisi**

Uygulama, sunucusuz (serverless) mimarisi ve istemci tarafı (client-side) JavaScript tabanlı bir yapı kullanır.

* **İstemci (Frontend):**
  + **HTML (index.html, login.html, signup.html, profile.html, settings.html, about.html):** Uygulamanın iskeletini ve sayfa yapılarını oluşturur.
  + **CSS (style.css):** Uygulamanın görsel stilini ve temalarını (chapter'lara göre değişen renkler) tanımlar.
  + **JavaScript (app.js):** Uygulamanın ana iş mantığını, DOM manipülasyonlarını, kullanıcı etkileşimlerini ve Firebase ile iletişimi yönetir.
* **Arka Uç (Backend - Serverless):**
  + **Firebase Authentication:** Kullanıcı kayıt, giriş ve oturum yönetimini sağlar.
  + **Firebase Firestore:** Kullanıcıların alışkanlıkları, ilerlemeleri ve profil bilgileri gibi verilerini depolayan NoSQL bulut veritabanıdır.
* **Bağlantı:** firebase.js dosyası, Firebase projenizin yapılandırma bilgilerini içerir ve Firebase servislerini başlatır. app.js bu başlatılmış servisleri kullanarak Firebase ile iletişim kurar.

**Mimari Diyagram (Basit):**

+----------------+ +-------------------------+ +---------------------+

| Kullanıcı Arayüzü | | JavaScript Mantığı | | Firebase |

| (HTML, CSS) | <-----> | (app.js) | <-----> | (Auth, Firestore) |

+----------------+ +-------------------------+ +---------------------+

^ | ^

| | |

| | |

+----------------------------+----------------------------------+

Web Tarayıcısı

**4. Teknik Tasarım Detayları**

**4.1. Veritabanı Yapısı (Firebase Firestore)**

Firestore'da temel olarak iki ana koleksiyon bulunmaktadır:

* **users Koleksiyonu:**
  + Her doküman bir kullanıcıyı temsil eder ve userId (Firebase UID) ile adlandırılır.
  + **Alanlar:**
    - email: (string) Kullanıcının e-posta adresi.
    - currentChapter: (number) Kullanıcının mevcut bölüm numarası (varsayılan: 1).
    - totalDaysCompleted: (number) Kullanıcının bugüne kadar tamamladığı toplam gün sayısı.
    - lastLoginDate: (string) Son giriş yapılan tarih (günlük sıfırlama için).
    - streak: (number) Mevcut başarılı seri (ardışık gün sayısı).
    - theme: (string) Kullanıcının seçtiği tema (örn: 'light', 'dark').
    - goalsAddedThisChapter: (number) Bu bölümde eklenen alışkanlık sayısı.
* **goals Koleksiyonu:**
  + Her doküman bir alışkanlığı temsil eder. Otomatik olarak oluşturulan bir ID'ye sahiptir.
  + **Alanlar:**
    - userId: (string) Alışkanlığın ait olduğu kullanıcının ID'si.
    - title: (string) Alışkanlığın başlığı.
    - description: (string) Alışkanlığın açıklaması.
    - icon: (string) Alışkanlığı temsil eden emoji veya ikon.
    - completedDays: (Array&lt;string>) Alışkanlığın tamamlandığı tarihlerin listesi (YYYY-MM-DD formatında).
    - isCompletedToday: (boolean) Alışkanlığın o gün için tamamlanıp tamamlanmadığı.
    - timestamp: (Timestamp) Alışkanlığın oluşturulma zamanı.

**4.2. Kimlik Doğrulama Akışı**

1. Kullanıcı login.html veya signup.html sayfasına gider.
2. Form verileri (e-posta, şifre) app.js tarafından alınır.
3. firebase/auth servisleri ( createUserWithEmailAndPassword veya signInWithEmailAndPassword ) ile Firebase'e istek gönderilir.
4. Başarılı olursa, onAuthStateChanged dinleyicisi tetiklenir.
5. GoalTracker sınıfı renderApp() metodunu çağırarak kullanıcıyı index.html'e yönlendirir ve kullanıcıya özel verileri (alışkanlıklar, profil) yükler.
6. Başarısız olursa, kullanıcıya hata mesajı gösterilir.

**4.3. Alışkanlık Yönetimi Akışı (Örnek: Alışkanlığı Tamamlandı İşaretleme)**

1. Kullanıcı index.html sayfasında belirli bir alışkanlığın onay işaretine tıklar.
2. app.js içindeki ilgili olay dinleyicisi tetiklenir.
3. Tıklanan alışkanlığın ID'si alınır.
4. Firestore'da goals koleksiyonundaki ilgili doküman güncellenir (updateDoc). isCompletedToday alanı true yapılır ve completedDays dizisine mevcut tarih eklenir.
5. Uygulamanın yerel goals dizisi güncellenir.
6. İlerleme çubuğu ve "Tamamlanan Günler" sayacı güncellenir.
7. Uygulama, günlük tamamlanan alışkanlık sayısına göre bölüm tamamlama koşullarını kontrol eder.
8. Eğer 21 günlük döngü tamamlandıysa, progressToNextChapter() metodu çağrılır. Bu metot, users koleksiyonundaki kullanıcının currentChapter değerini artırır ve tüm alışkanlıkların isCompletedToday durumunu sıfırlar.

**4.4. Tema Yönetimi**

* settings.html sayfasındaki bir <select> elementi ile kullanıcı tema seçimi yapabilir.
* Seçilen tema localStorage'a kaydedilir.
* app.js içindeki applyTheme() fonksiyonu, body etiketine ilgili tema sınıfını (örn: light-theme, dark-theme) ve chapter-X sınıfını ekleyerek CSS değişkenlerini (var(--primary-color)) günceller.

**5. Test ve Doğrulama**

* **Birim Testleri:** Her bir JavaScript fonksiyonunun (örn: addNewGoal, markGoalCompleted) bağımsız olarak doğru çalışıp çalışmadığını doğrulamak için testler yazılabilir.
* **Entegrasyon Testleri:** Firebase ile olan etkileşimlerin (veri kaydetme, okuma, güncelleme) doğru çalıştığından emin olmak için testler yapılabilir.
* **Kullanıcı Kabul Testleri (UAT):** Uygulamanın son kullanıcı beklentilerini karşıladığından emin olmak için hedef kitle ile testler yapılabilir.
* **Performans Testleri:** Uygulamanın farklı yük altında nasıl davrandığını ölçmek için testler yapılabilir.

**6. Dağıtım (Deployment)**

Uygulama, HTML, CSS ve JavaScript'ten oluşan statik bir web uygulamasıdır. Bu nedenle, Firebase Hosting, Netlify, Vercel veya GitHub Pages gibi statik site barındırma hizmetleri kullanılarak kolayca dağıtılabilir. Firebase entegrasyonu sayesinde backend operasyonları otomatik olarak yönetilir.