



softlTo 4. DÖNEM 1. GRUP
FRONTEND DEVELOPER PROGRAMI

BİREYSEL BİTİRME PROJESİ RAPORU

MyVid AI

Tarayıcı Tabanlı Yapay Zeka Destekli Video Analiz ve Annotation Uygulaması

Hazırlayan	Ünal Öztürk
Proje Danışmanı	Selahaddin Çiftçi
Proje Dönemi	6 Temmuz - 13 Temmuz 2026

Temmuz 2026
softlTo Yazılım Bilişim Akademisi, İstanbul

ÖZET

MyVid AI; kullanıcının kendi video dosyalarını tarayıcı içinde açmasına, videodaki kişi ve nesneleri kutularla işaretlemesine, yapay zeka destekli nesne tespiti uygulamasına, seçilen nesnenin hareketini zaman boyunca takip etmesine ve ilgili bölümleri dışa aktarmasına olanak sağlayan istemci taraflı bir video analiz uygulamasıdır. Proje, video dosyasının zorunlu olarak bir uygulama sunucusuna yüklenmediği yerel-öncelikli bir mimariyle geliştirilmiştir. Büyük video verileri IndexedDB içinde, proje ve annotation bilgileri ise Redux Persist aracılığıyla tarayıcı depolamasında saklanmaktadır.

Uygulamanın kullanıcı arayüzü React ve Vite ile oluşturulmuş; merkezi durum yönetimi Redux Toolkit, etkileşimli çizim alanı React Konva, nesne tespiti MediaPipe Object Detector, video kesme işlemleri FFmpeg WebAssembly ve isteğe bağlı sahne yorumlama Gemini API ile gerçekleştirilmiştir. Annotation kutuları göreceli koordinatlarla saklanarak farklı ekran boyutlarında video üzerindeki konumların korunması amaçlanmıştır. Proje yönetimi, özel video kontrolleri, zaman çizelgesi, JSON dışa aktarma ve Vercel üzerinde statik dağıtım uygulamanın bütünsel parçalarıdır.

Bu raporda projenin amacı, kapsamı, kullanılan teknolojiler, sistem mimarisi, veri modeli, temel algoritmaları, kullanıcı akışları, gizlilik yaklaşımı, dağıtım yapısı ve doğrulama sonuçları ayrıntılı olarak sunulmuştur. Sonuç olarak MyVid AI, video inceleme ve annotation işlemlerini tek bir modern web arayüzünde birleştiren; sunucu bağımlılığını azaltan ve kullanıcının verisi üzerinde kontrolünü önceleyen işlevsel bir frontend bitirme projesi ortaya koymaktadır.

Anahtar Kelimeler: video analizi, annotation, React, MediaPipe, nesne tespiti, nesne takibi, FFmpeg.wasm, Gemini, IndexedDB

ABSTRACT

MyVid AI is a client-side video analysis application that enables users to open local video files in a browser, annotate people and objects with bounding boxes, apply AI-assisted object detection, follow a selected object's movement over time, and export relevant outputs. The project follows a local-first architecture in which the original video is not required to be uploaded to an application server. Large video files are stored in IndexedDB, while project metadata and annotation records are persisted through Redux Persist.

The user interface is built with React and Vite. Redux Toolkit manages application state, React Konva provides the interactive canvas layer, MediaPipe Object Detector performs local object detection, FFmpeg WebAssembly handles browser-side trimming, and the Gemini API provides optional scene interpretation. Relative coordinates preserve annotation placement across responsive layouts. The resulting system demonstrates an integrated frontend solution for project management, video playback, annotation, detection, tracking, timeline navigation, and JSON export.

Keywords: video analysis, annotation, React, MediaPipe, object detection, object tracking, FFmpeg.wasm, Gemini, IndexedDB

İÇİNDEKİLER

Şekiller Listesi	5
Tablolar Listesi	5
Kısaltmalar	5
1. GİRİŞ	6
1.1. Problem Tanımı	6
1.2. Projenin Amacı	6
1.3. Kapsam ve Kısıtlar	6
2. PROJE HEDEFLERİ VE GELİŞTİRME YÖNTEMİ	7
2.1. Fonksiyonel Gereksinimler	7
2.2. Fonksiyonel Olmayan Gereksinimler	7
3. MATERYAL, TEKNOLOJİ VE ARAÇLAR	7
3.1. React ve Vite	8
3.2. Redux Toolkit ve Kalıcı Durum	8
3.3. MediaPipe, FFmpeg ve Gemini	8
4. SİSTEM MİMARİSİ	8
4.1. Uygulama Rotaları	9
4.2. Kullanıcı Akışı	9
4.3. Klasör ve Modül Organizasyonu	9
5. VERİ MODELİ VE DURUM YÖNETİMİ	10
5.1. Proje Verisi	10
5.2. Annotation Verisi	10
5.3. Redux Slice Yapısı	11
6. UYGULAMANIN GERÇEKLEŞTİRİLMESİ	11
6.1. Karşılama ve Dashboard	11
6.2. Proje Oluşturma ve Thumbnail	11
6.3. Video Oynatıcı ve Kontroller	11
6.4. Annotation Canvas	11
6.5. Yapay Zeka ile Nesne Tespiti	12
6.6. Akıllı Nesne Takibi	13
6.7. Timeline	13
6.8. FFmpeg.wasm ile Kesit Oluşturma	13
6.9. Gemini ile Storyboard Analizi	14
6.10. JSON Dışa Aktarma	14

7. KULLANICI ARAYÜZÜ VE DENEYİM TASARIMI	14
7.1. Çalışma Alanı Yerleşimi	14
7.2. Responsive Yaklaşım	14
7.3. Kullanılabilirlik Kararları	14
8. GİZLİLİK VE GÜVENLİK YAKLAŞIMI	15
8.1. Tarayıcı İzolasyon Başlıkları	15
8.2. Kullanıcı Sorumluluğu ve Şeffaflık	15
9. KURULUM VE DAĞITIM	15
9.1. Geliştirme Ortamı	15
9.2. Vercel Dağıtımı	16
9.3. Çalışma Koşulları	16
10. TEST VE DOĞRULAMA	16
10.1. Bağımlılık ve Kod Kalitesi	17
10.2. Tarayıcı ve Format Değerlendirmesi	17
10.3. Performans Yaklaşımı	17
11. SINIRLAMALAR VE GELECEK ÇALIŞMALAR	17
11.1. Önerilen Geliştirmeler	17
12. SONUÇ	18
13. KAYNAKÇA	19

ŞEKİLLER, TABLOLAR VE KISALTMALAR

Şekiller Listesi

Şekil 4.1. MyVid AI genel sistem mimarisi

Şekil 4.2. Temel kullanıcı çalışma akışı

Şekil 5.1. Tarayıcı tabanlı veri saklama modeli

Şekil 6.1. Relative annotation koordinat sistemi

Şekil 6.2. Nesne takip işlem akışı

Tablolar Listesi

Tablo 2.1. Proje hedefleri

Tablo 3.1. Temel teknoloji ve bağımlılıklar

Tablo 4.1. Uygulama rotaları

Tablo 5.1. Redux veri modeli

Tablo 8.1. Gizlilik ve veri hareketi

Tablo 10.1. İşlevsel doğrulama matrisi

Tablo 10.2. Tarayıcı ve format değerlendirmesi

Kısaltmalar

AI	Artificial Intelligence - Yapay Zeka
API	Application Programming Interface
COEP	Cross-Origin Embedder Policy
COOP	Cross-Origin Opener Policy
FPS	Frames Per Second - Saniyedeki kare sayısı
IoU	Intersection over Union - Kesişim / birleşim oranı
SPA	Single Page Application
UI/UX	Kullanıcı arayüzü / kullanıcı deneyimi
WASM	WebAssembly

1. GİRİŞ

Video içerikleri; eğitim, içerik üretimi, araştırma, spor analizi, güvenlik incelemesi ve veri etiketleme gibi çok sayıda alanda kullanılmaktadır. Ancak uzun bir video içinde belirli bir nesneyi bulmak, görünür olduğu anları işaretlemek ve ilgili bölümü ayırmak çoğu zaman ayrı araçlar gerektirir. Geleneksel video düzenleme yazılımları güçlü olmakla birlikte yalnızca annotation ve temel analiz yapmak isteyen kullanıcılar için karmaşık olabilir. Bulut tabanlı çözümlerde ise büyük video dosyalarının yüklenmesi zaman, bant genişliği ve gizlilik açısından ek maliyet oluşturur.

MyVid AI bu probleme, modern tarayıcı yeteneklerini kullanan tek sayfalı bir web uygulamasıyla yaklaşmaktadır. Kullanıcı bir MP4 veya WebM videosunu seçerek proje oluşturur; video cihazdaki tarayıcı veritabanında saklanır ve Blob URL üzerinden oynatılır. Kullanıcı videoyu hassas zaman kontrolleriyle inceleyebilir, kare üzerine kutu çizebilir, yerel nesne tespitiinden yararlanabilir, seçilen kutu için hareket rotası üretebilir ve sonuçları JSON ya da video kesiti biçiminde dışa aktarabilir.

1.1. Problem Tanımı

Temel problem, video inceleme, nesne işaretleme, basit takip, sahne yorumlama ve kesit oluşturma işlemlerinin tek bir erişilebilir arayüzde birleştirilmesidir. Çözümün büyük video dosyalarını uygulama sunucusuna taşımadan çalışması; annotation koordinatlarını farklı ekran boyutlarında koruması; kullanıcının proje verisini tarayıcı oturumları arasında sürdürebilmesi ve yapay zeka özelliklerini isteğe bağlı sunması beklenmiştir.

1.2. Projenin Amacı

- Kullanıcının yerel MP4 ve WebM videolarıyla proje oluşturabilmesini sağlamak.
- Video oynatma, zaman çizelgesi, hız ve kareye yakın adım kontrolleri sunmak.
- Video üzerine seçilebilir, taşınabilir ve yeniden boyutlandırılabilir kutular çizmek.
- MediaPipe ile geçerli karedeki nesneleri tarayıcı içinde tespit etmek.
- Seçili nesne için zaman tabanlı keyframe dizisi ve görsel hareket rotası oluşturmak.
- İlgili zaman aralığını FFmpeg.wasm ile istemci tarafında kesmek.
- İsteğe bağlı olarak storyboard karelerini Gemini ile Türkçe yorumlamak.
- Annotation verisini açık ve taşınabilir JSON biçiminde dışa aktarmak.

1.3. Kapsam ve Kısıtlar

Proje tek kullanıcı ve tarayıcı tabanlı bir çalışma alanı olarak tasarlanmıştır. Kullanıcı hesabı, ekip çalışması, bulut senkronizasyonu ve sunucu taraflı video depolama kapsam dışındadır. Nesne takibi profesyonel re-identification modelleri yerine ardışık tespitler arasındaki mekansal yakınlığı kullanan anlaşılır bir yaklaşım uygular. Video kesme başarımı kaynak codec ile tarayıcı belleğine; yerel yapay zeka özellikleri cihazın WebAssembly ve WebGL desteğine bağlıdır. Bu sınırlar projenin frontend odaklı, kurulumu kolay ve gizlilik bilinci yüksek kalmasını sağlamıştır.

Projenin özgün değeri

MyVid AI'nin değeri tek bir algoritmadan değil; proje yönetimi, yerel video saklama, etkileşimli canvas annotation, yapay zeka destekli tespit, temel takip, kesme ve sahne analizi bileşenlerini tutarlı bir kullanıcı akışında birleştirmesinden oluşmaktadır.

2. PROJE HEDEFLERİ VE GELİŞTİRME YÖNTEMİ

Geliştirme süreci 6-13 Temmuz 2026 tarihleri arasında kısa ve odaklı bir ürün geliştirme döngüsü olarak yürütülmüştür. Önce kullanıcı yolculuğu ve ekranlar belirlenmiş, ardından veri saklama ve durum yönetimi kurulmuş, video/annotation etkileşimi geliştirilmiş ve yapay zeka ile dışa aktarma servisleri bütünleştirilmiştir. Bileşen tabanlı tasarım sayesinde ekranlar, servisler ve Redux dilimleri birbirinden ayrılmıştır.

Yerel video yönetimi	Video sunucuya zorunlu yüklenmeden açılabilmesi	IndexedDB + Blob URL
Hassas inceleme	Zaman, hız ve ileri/geri kontrolleri	Özel VideoControls
Etkileşimli annotation	Kutular çizilebilmesi ve düzenlenebilmesi	React Konva Stage/Layer
AI tespiti	Geçerli karede kutu ve etiket üretilebilmesi	MediaPipe Object Detector
Hareket takibi	Koordinatlar zaman dizisi olarak saklanmalı	Keyframe tabanlı rota
Çıktı	JSON ve video kesiti alınabilmesi	Blob indirme + FFmpeg

Tablo 2.1. Proje hedefleri ve uygulamadaki karşılıkları

2.1. Fonksiyonel Gereksinimler

- Karşılama sayfası, dashboard ve annotation çalışma alanı arasında yönlendirme.
- Yeni proje oluşturma, proje adını düzenleme, projeyi açma ve silme.
- Videodan otomatik küçük önizleme görseli üretme.
- Video oynatma, duraklatma, seek, hız seçimi ve zaman gösterimi.
- Dikdörtgen annotation çizme, seçme, taşıma, yeniden boyutlandırma ve etiketleme.
- AI ile nesne tespiti, IoU ile tekrar kutularını azaltma ve seçilen nesneyi takip etme.
- Timeline üzerinde annotation ve keyframe işaretlerini gösterme.
- Gemini anahtarını kullanıcıdan alma ve sahne analizi gerçekleştirme.
- Annotation JSON'u ve takip edilen video bölümünü indirme.

2.2. Fonksiyonel Olmayan Gereksinimler

- Modern masaüstü ve mobil ekranlara uyumlu responsive arayüz.
- Büyük video verilerini localStorage yerine IndexedDB'de tutma.
- Kullanıcı videosunun temel işlemlerde yerel kalması.
- Yoğun servisleri ihtiyaç anında başlatma ve kullanıcıya ilerleme durumu gösterme.
- SPA alt rotalarının Vercel üzerinde doğrudan açılabilmesi.
- Sade, tutarlı ve tekrar kullanılabilir UI bileşenleri.

3. MATERYAL, TEKNOLOJİ VE ARAÇLAR

MyVid AI, frontend geliştirme programının kazanımlarını modern tarayıcı API'leri ve yapay zeka servisleriyle birleştirmektedir. package-lock dosyası bağımlılık ağacını sabitlerken Vite geliştirme ve üretim süreçlerini yönetir. Aşağıdaki tabloda raporun hazırlandığı kaynak kodda kullanılan temel teknolojiler verilmiştir.

React / React DOM	19.2.7	Bileşen tabanlı kullanıcı arayüzü

Vite	8.1.x	Geliştirme sunucusu ve production build
React Router DOM	7.18.1	SPA yönlendirme
Redux Toolkit	2.12.0	Merkezi proje ve annotation durumu
Redux Persist	6.0.0	Redux state kalıcılığı
idb	8.0.3	Promise tabanlı IndexedDB erişimi
Konva / React Konva	10.3.0 / 19.2.5	Canvas çizim ve dönüştürme işlemleri
MediaPipe Tasks Vision	0.10.35	Yerel nesne tespiti
FFmpeg.wasm	0.12.x	Tarayıcı içinde video kesme
Google Generative AI	0.24.1	Gemini storyboard analizi
Tailwind CSS	4.3.2	Tema ve responsive stiller
Base UI / shadcn	1.6.0 / 4.13.0	Erişilebilir ortak arayüz bileşenleri

Tablo 3.1. Temel teknoloji ve bağımlılıklar

3.1. React ve Vite

React ekranları fonksiyon bileşenleri ve hook'larla yönetir. Uygulamanın giriş noktası StrictMode altında App bileşenini oluşturur; App ise Redux Provider, PersistGate, RouterProvider ve bildirim katmanını bir araya getirir. Vite; JSX dönüştürme, hızlı geliştirme sunucusu, Tailwind eklentisi ve statik production çıktısı sağlar. Bu seçim, kısa proje takviminde hızlı geliştirme ile modüler kod organizasyonunu desteklemiştir.

3.2. Redux Toolkit ve Kalıcı Durum

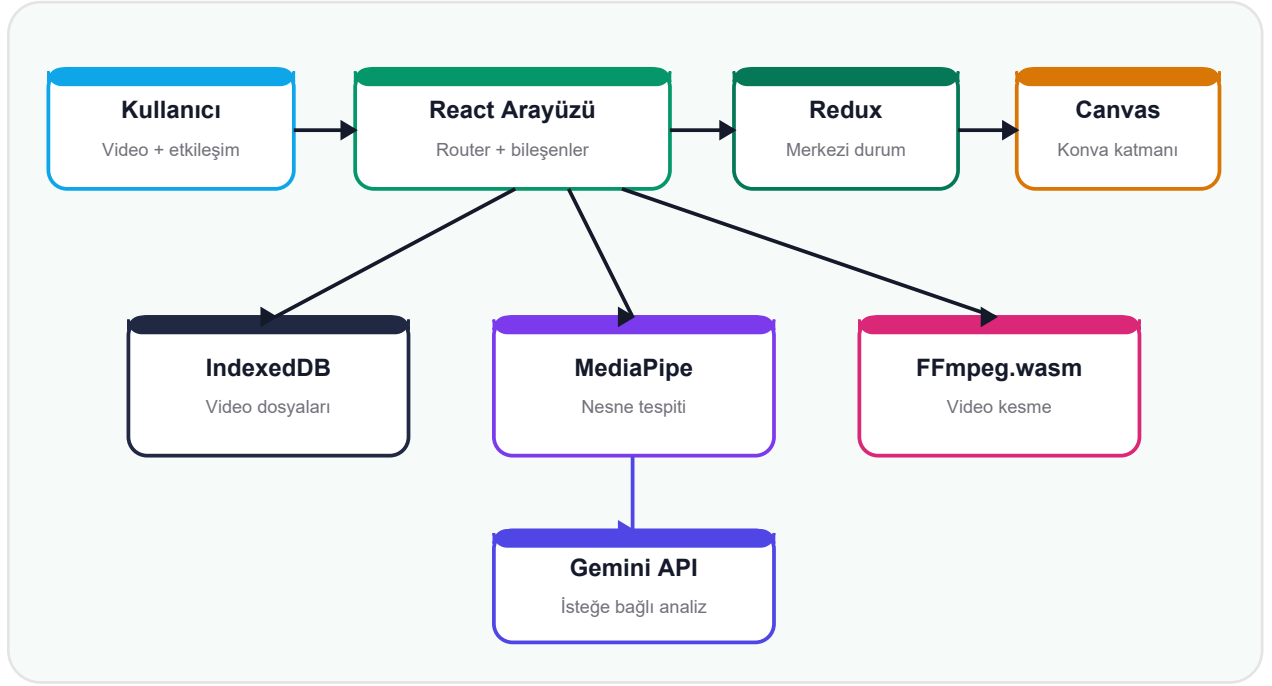
Projeler, annotation'lar ve ayarlar ayrı Redux slice'larıdır. createSlice yaklaşımı reducer ve action tanımlarını tek yerde toplar. Redux Persist seçili slice'ları tarayıcı depolamasına yazar; böylece sayfa yenilendiğinde proje metadata'sı ve annotation çalışması korunur. Video dosyalarının Redux içine alınmaması, serializable state ve localStorage boyutu açısından bilinçli bir ayrımdır.

3.3. MediaPipe, FFmpeg ve Gemini

MediaPipe Object Detector, görüntü içindeki nesneler için sınıf, güven skoru ve bounding box üretir. MyVid AI EfficientDet Lite2 modelini GPU delegesiyle başlatmayı dener ve IMAGE modunda kareleri bağımsız analiz eder. FFmpeg.wasm, klasik FFmpeg işlevlerini WebAssembly üzerinden tarayıcıya taşır; uygulama seçilen zaman aralığını sanal dosya sistemi içinde işler. Gemini ise yalnızca kullanıcı sahne analizini başlattığında, geçerli zamanın çevresinden çıkarılan sınırlı sayıda JPEG kareyi multimodal istek içinde yorumlar.

4. SİSTEM MİMARİSİ

Uygulama backend zorunluluğu olmayan bir SPA olarak çalışır. Kullanıcı etkileşimleri React bileşenlerinde karşılanır; Redux paylaşılan durumu yönetir; IndexedDB büyük video dosyalarını saklar; Konva video üzerinde etkileşimli çizim katmanı oluşturur. AI ve medya servisleri kullanıcı işlemleriyle tetiklenen bağımsız modüllerdir.



Şekil 4.1. MyVid AI genel sistem mimarisi

4.1. Uygulama Rotaları

/	WelcomeScreen	Ürün tanıtımı, özellikler ve uygulamaya geçiş
/app	DashboardScreen	Proje listesi, yeni proje ve proje kartları
/app/projects/:projectId/annotate	AnnotationWorkspace	Video, canvas, araçlar, timeline ve Gemini paneli

Tablo 4.1. Uygulama rotaları

4.2. Kullanıcı Akışı



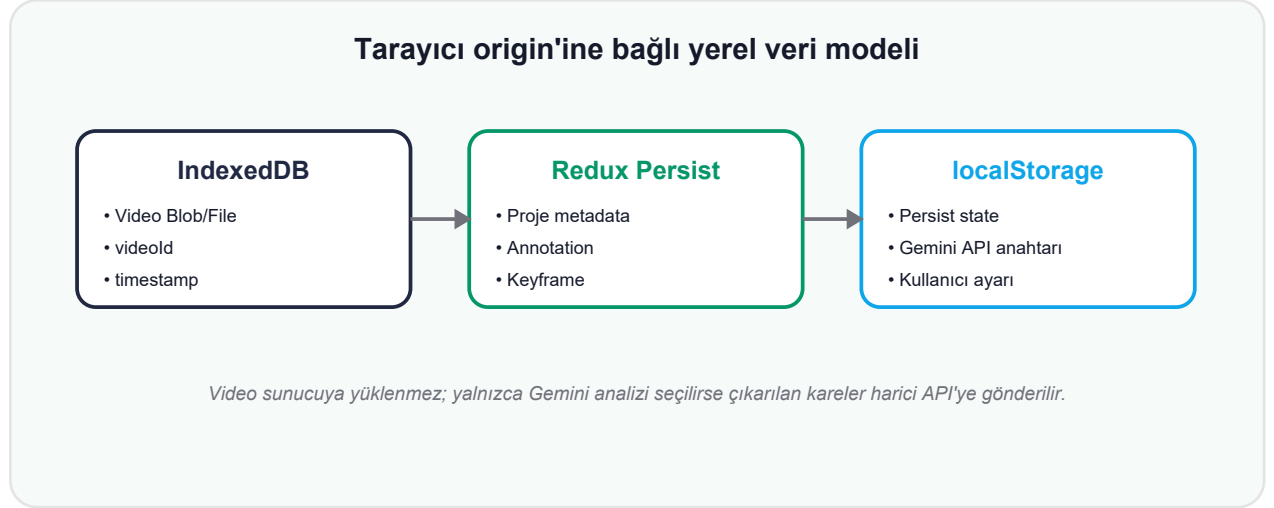
Şekil 4.2. Temel kullanıcı çalışma akışı

4.3. Klasör ve Modül Organizasyonu

Kaynak kod, özellik odaklı klasörleme ile düzenlenmiştir. app klasörü store ve router yapılandırmasını; services klasörü MediaPipe, FFmpeg, Gemini ve IndexedDB erişimini; utils klasörü geometri ve video yardımcılarını; features klasörü ise kullanıcıya sunulan ekran ve işlevleri içerir. Ortak UI bileşenleri components/ui altında tutulur. Böylece görsel bileşen, uygulama durumu ve dış servis sorumlulukları ayrıştırılır.

```
src/
  app/                router.jsx, store.js
  components/ui/      button, dialog, slider, card, toaster...
  features/
    projects/         proje oluşturma ve kartlar
    video-player/     video ve özel kontroller
    annotations/      canvas, timeline, araçlar, Gemini paneli
  services/           ai, database, ffmpeg, gemini
  utils/              geometry, videoUtils
  styles/             global tema ve feature stilleri
```

5. VERİ MODELİ VE DURUM YÖNETİMİ



Şekil 5.1. Tarayıcı tabanlı veri saklama modeli

5.1. Proje Verisi

Yeni proje için zaman damgasına dayalı bir kimlik oluşturulur. Video aynı kimlikle IndexedDB'deki videos object store'una yazılır. Videodan çıkarılan küçük JPEG thumbnail, proje adı, videoid ve zaman bilgileri Redux projects listesine eklenir. Dashboard kartları bu metadata üzerinden oluşturulur; video dosyasının kendisi Redux state içinde tutulmaz.

```
{
  id: "proj_...",
  name: "Proje adı",
  videoId: "proj_...",
  thumbnail: "data:image/jpeg;base64,...",
  createdAt: 178..., updatedAt: 178...
}
```

5.2. Annotation Verisi

Annotation'lar proje kimliğine göre gruplanır. Her kayıt benzersiz id, görelî kutu koordinatları, renk, etiket ve isteğe bağlı keyframe dizisi taşır. Keyframe, belirli bir zamanda nesnenin görelî kutusunu temsil eder. Bu yapı video oynatılırken iki kayıt arasında doğrusal interpolasyon yapılmasına olanak verir.

```
{
  id: "annotation-id",
  rx: 0.25, ry: 0.18,
  rwidth: 0.20, rheight: 0.32,
  color: "#f59e0b", label: "person",
  keyframes: [
    { time: 1.50, rx: 0.25, ry: 0.18,
      rwidth: 0.20, rheight: 0.32 }
  ]
}
```

5.3. Redux Slice Yapısı

projects	list	addProject, updateProject, deleteProject
annotations	itemsByProject, activeTool, selectedId, tracking state	add/update/delete annotation, keyframe ve seçim
settings	theme, autoSave	Kullanıcı ayarlarını temsil eden altyapı

Tablo 5.1. Redux veri modeli

6. UYGULAMANIN GERÇEKLEŞTİRİLMESİ

6.1. Karşılama ve Dashboard

Karşılama ekranı ürünün yerel video işleme, akıllı annotation ve AI sahne analizi özelliklerini tanıtır. Responsive kartlar ve üç aşamalı çalışma akışı kullanıcıyı uygulamaya hazırlar. Dashboard, Redux'ta saklanan projeleri kartlar halinde listeler; kullanıcı proje adını kart üzerinde düzenleyebilir, projeyi açabilir veya onay diyaloguyla silebilir. Boş durum mesajı ilk kez gelen kullanıcıyı yeni proje oluşturmaya yönlendirir.

6.2. Proje Oluşturma ve Thumbnail

ProjectWizard diyalogu proje adı ve MP4/WebM dosyası ister. Gönderim sırasında video önce IndexedDB'ye kaydedilir. Ardından geçici bir HTMLVideoElement oluşturulur, uygun bir zamana seek edilir ve kare canvas üzerine çizilerek en fazla 320 piksel genişliğinde JPEG thumbnail üretilir. Object URL işlem sonunda iptal edilir. Metadata Redux'a eklendikten sonra kullanıcı doğrudan annotation rotasına yönlendirilir.

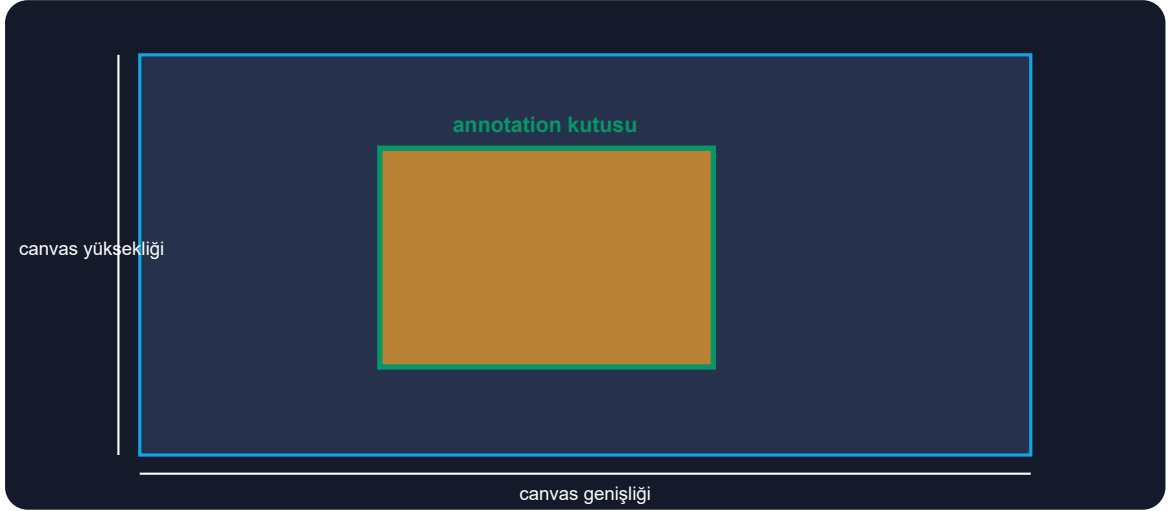
6.3. Video Oynatıcı ve Kontroller

VideoPlayer, IndexedDB'den alınan File/Blob için object URL oluşturur ve HTML video elemanına bağlar. VideoControls bileşeni play, pause, bir saniye ileri/geri, yaklaşık kare adımı, oynatma hızı ve slider tabanlı seek sağlar. timeupdate, loadedmetadata, durationchange, ratechange, play ve pause olayları React state ile senkronize edilir. Oynatma sırasında requestAnimationFrame, zaman göstergesinin akıcı güncellenmesini destekler.

6.4. Annotation Canvas

React Konva Stage ve Layer, video alanının üzerinde şeffaf bir çizim yüzeyi oluşturur. Rect aracıyla fare veya dokunma hareketinden başlangıç ve bitiş noktaları alınır. Pointer aracı kutuyu seçer; Transformer yeniden boyutlandırma tutamaçlarını sağlar. Seçili annotation'ın etiketi üst araç çubuğundan değiştirilebilir; Delete/Backspace veya silme aracı kayıt üzerinde işlem yapar.

Relative koordinatlar ekran boyutundan bağımsız saklanır



Şekil 6.1. Relative annotation koordinat sistemi

Kutular mutlak piksel yerine canvas boyutuna bölünmüş oranlarla saklanır. Gösterim sırasında oranlar güncel genişlik ve yükseklikle yeniden çarpılır. Bu yaklaşım responsive tasarımda kutunun video üzerindeki görelî konumunu korur. MediaPipe bounding box'ları video çözünürlüğünden canvas'a aktarılırken object-fit: contain ölçeği ve letterbox boşlukları hesaba katılır.

```
rx      = x / canvasWidth
ry      = y / canvasHeight
rwidth  = width / canvasWidth
rheight = height / canvasHeight

x      = rx * currentCanvasWidth
y      = ry * currentCanvasHeight
```

6.5. Yapay Zeka ile Nesne Tespiti

AIService ilk kullanımda MediaPipe Vision WASM dosyalarını ve EfficientDet Lite2 modelini yükler. GPU delegeesi destekleniyorsa WebGL üzerinden hızlandırma kullanılır. Model IMAGE modunda çalışır ve 0,30 güven eşiğinin altındaki sonuçları eler. detect() sonucundaki bounding box ve category bilgileri uygulama koordinat modeline dönüştürülür. Otomatik kutular amber renk ile manuel annotation'lardan görsel olarak ayrılır.

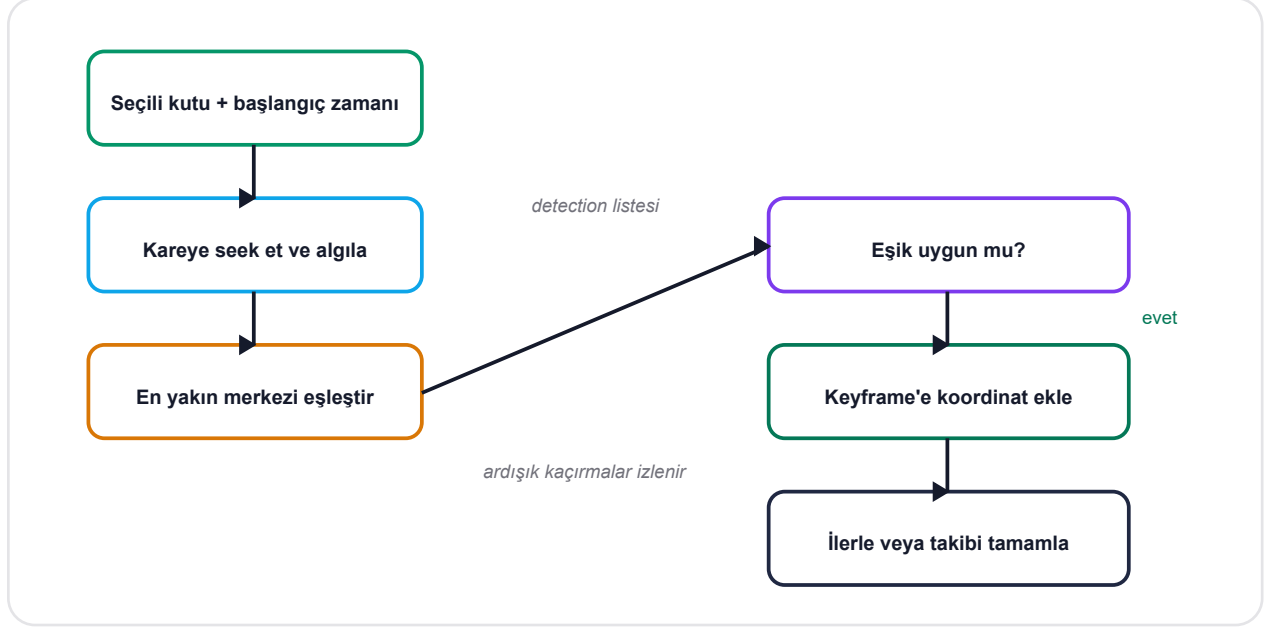
Aynı nesne için çok sayıda üst üste kutu eklenmesini azaltmak amacıyla yeni detection ile mevcut annotation'lar arasında Intersection over Union hesaplanır. Kesişim alanının birleşim alanına oranı 0,60'tan büyükse detection tekrar kabul edilerek eklenmez. Böylece AI işlemi mevcut kullanıcı çalışmasını tamamlayan bir yardımcı olarak davranır.

```
IoU = intersectionArea /
      (box1Area + box2Area - intersectionArea)

if (calculateIoU(mappedBox, annotation) > 0.60) {
  // Aynı bölgede bulunan tekrar kutusunu ekleme
}
```

6.6. Akıllı Nesne Takibi

Takip işlemi seçili annotation'ın kutusunu başlangıç kabul eder. Video duraklatılır; süreye göre 0,15, 0,25 veya 0,40 saniyelik tarama aralığı seçilir. Her adımda ilgili kare tekrar kullanılabilir canvas'a çizilir ve MediaPipe ile analiz edilir. Önceki kutu merkezine en yakın detection, belirlenen mesafe eşiğini sağlıyorsa yeni keyframe olarak kaydedilir. Art arda kaçırılan kareler takip durumunu kontrollü biçimde sonlandırır.



Şekil 6.2. Nesne takip işlem akışı

Oynatma sırasında annotation kutusu iki keyframe arasındaki zamansal ilerlemeye göre x, y, genişlik ve yükseklik değerlerini doğrusal biçimde interpolasyon eder. Konva node değerleri requestAnimationFrame içinde doğrudan güncellenerek her video zamanı için Redux dispatch yapılmasının önüne geçilir. Takip yüzdesi merkezi overlay ve araç çubuğu üzerinden kullanıcıya gösterilir.

6.7. Timeline

TimelinePanel her annotation için ayrı bir track oluşturur. Track etiketi, renk göstergesi ve keyframe işaretleri içerir. İşaret seçildiğinde video ilgili zamana taşınır ve annotation seçili hale gelir. Oynatma kafası video currentTime değerinin toplam süreye oranı üzerinden hareket eder. Seçilen track'in görünür alana kaydırılması yoğun annotation listelerinde kullanım kolaylığı sağlar.

6.8. FFmpeg.wasm ile Kesit Oluşturma

Kesme işlemi yalnızca keyframe dizisi bulunan takip edilmiş annotation için sunulur. İlk ve son keyframe zamanları başlangıç ve bitiş aralığını belirler. Video Blob'u FFmpeg sanal dosya sistemine yazılır; -ss, -t ve stream copy parametreleriyle MP4 çıktı üretilir. Sonuç Blob URL olarak indirilir ve geçici URL iptal edilir. İşlem yüzdesi toast ve araç ikonu üzerinden izlenebilir.

```

await ffmpeg.exec([
  "-ss", startTime,
  "-i", "input.mp4",
  "-t", duration,
  "-c", "copy",
  "output.mp4"
])

```

6.9. Gemini ile Storyboard Analizi

Gemini özelliği uygulamanın diğer işlevlerinden bağımsız ve isteğe bağlıdır. Kullanıcı kendi API anahtarını girer. Geçerli video zamanının bir saniye öncesi, kendisi ve bir saniye sonrasında mümkün olan benzersiz kareler çıkarılır. Kareler yarı çözünürlükte JPEG'e dönüştürülür ve base64 inlineData parçaları olarak Gemini 3.5 Flash modeline gönderilir. Prompt; sahnedeki kişiler/nesneler, hareket ve genel eylem için kısa Türkçe açıklama ister.

İncelenen kareler panelde küçük önizlemelerle gösterilir; model yanıtı whitespace korunarak okunabilir bir kart içinde sunulur. API anahtarı uygulama sunucusuna kaydedilmez, tarayıcı localStorage alanında tutulur ve kullanıcı tarafından temizlenebilir.

6.10. JSON Dışa Aktarma

Çalışma alanındaki annotation dizisi girintili JSON metnine dönüştürülür, application/json türünde Blob oluşturulur ve proje adından türetilen dosya adıyla indirilir. Orijinal video JSON içine eklenmez. Çıktı; arşivleme, başka sistemlerde işleme veya annotation çalışmasının incelenmesi için açık bir veri biçimi sağlar.

7. KULLANICI ARAYÜZÜ VE DENEYİM TASARIMI

Arayüz açık zemin, emerald ana renk ve koyu video viewport kombinasyonuyla tasarlanmıştır. Tema değişkenleri merkezi CSS dosyasında tanımlanır. Tailwind utility sınıfları ile özel myvid-* bileşen sınıfları birlikte kullanılır. Card, Button, Dialog, AlertDialog, Input, Slider ve Toaster gibi ortak bileşenler Base UI/shadcn yapısı üzerinde tekrar kullanılabilir hale getirilmiştir.

7.1. Çalışma Alanı Yerleşimi

- Sol araç rayı: dashboard, seçim, çizim, AI tespit, takip, kesme ve silme araçları.
- Merkez panel: proje adı, video/canvas, playback kontrolleri ve annotation timeline.
- Sağ panel: Gemini API anahtarı, analiz komutu, storyboard kareleri ve AI yorumu.
- Takip overlay'i: video üzerinde merkezi ilerleme göstergesi ve durum metni.
- Toast katmanı: başarı, hata, bilgi ve yüklenme geri bildirimi.

7.2. Responsive Yaklaşım

Masaüstünde üç sütunlu çalışma alanı, küçük ekranlarda dikey akışa dönüşür. Sol ray yatay araç çubuğu haline gelir; Gemini paneli merkezin altına geçer. Dashboard kolon sayısı ekran genişliğine göre birden beşe kadar değişir. Video alanı aspect-video oranını korur; canvas ölçüleri pencere yeniden boyutlandığında güncellenir. 2XL sınıfları büyük ekran ve akıllı tahta senaryolarında yazı ve kontrol boyutlarını artırır.

7.3. Kullanılabilirlik Kararları

- Araçlar ikon ve tooltip birlikteliğiyle sunulur.
- AI ile eklenen kutular manuel kutulardan farklı renktedir.
- Seçili annotation için etiket alanı bağlama duyarlı görünür.
- Tehlikeli proje silme işlemi onay diyalogu kullanır.
- Uzun işlemlerde butonlar devre dışı kalır ve ilerleme gösterilir.
- Boş durum, hata ve yüklenme mesajları kullanıcıyı yönlendirir.

8. GİZLİLİK VE GÜVENLİK YAKLAŞIMI

MyVid AI'nin temel gizlilik kararı, orijinal videonun uygulama backend'ine yüklenmesini gerektirmemektir. Video IndexedDB'de tarayıcı origin'ine bağlı tutulur; oynatma, annotation, MediaPipe tespiti ve FFmpeg kesme istemci tarafında gerçekleşir. Bu yaklaşım özellikle kişisel veya henüz yayımlanmamış videolar üzerinde çalışırken veri hareketini azaltır.

Orijinal video	IndexedDB	Temel akışta paylaşılmaz
Proje ve annotation	Redux Persist / localStorage	Paylaşılmaz
Thumbnail	Proje metadata'sı içinde	Paylaşılmaz
MediaPipe kare analizi	Tarayıcı belleği	Model yürütmesi yereldir
Gemini storyboard kareleri	Geçici component state	Kullanıcı analiz başlatırsa Gemini API'ye gönderilir
Gemini API anahtarı	localStorage	İstek sırasında doğrudan Gemini API için kullanılır

Tablo 8.1. Gizlilik ve veri hareketi

8.1. Tarayıcı İzolasyon Başlıkları

Geliştirme sunucusu ve Vercel dağıtımı Cross-Origin-Opener-Policy: same-origin ile Cross-Origin-Embedder-Policy: require-corp başlıklarını ekler. Bu başlıklar WebAssembly ve tarayıcı izolasyonu gerektiren işlemler için uygun bir temel sağlar. React Router alt rotaları Vercel rewrite kuralıyla index.html dosyasına yönlendirilir.

8.2. Kullanıcı Sorumluluğu ve Şeffaflık

Gemini entegrasyonu BYOK - Bring Your Own Key yaklaşımıdır. Anahtarın localStorage içinde tutulduğu ve analiz sırasında karelerin harici servise gönderildiği açıkça belirtilmelidir. Kullanıcı tarayıcı verilerini temizlediğinde yerel projelerin kaybolabileceği için önemli annotation çalışmalarının JSON olarak dışa aktarılması önerilir. Bu açıklamalar kullanıcıya veri üzerinde bilinçli karar verme olanağı sağlar.

Güvenlik sonucu

Mimari, büyük video verisini uygulama sunucusundan uzak tutarak veri minimizasyonunu destekler. Harici yapay zeka yalnızca kullanıcı isteğiyle devreye girer; diğer temel özellikler API anahtarı olmadan çalışır.

9. KURULUM VE DAĞITIM

9.1. Geliştirme Ortamı

Proje Node.js ve npm bulunan bir ortamda package-lock ile tekrar kurulabilir. npm ci kilitli sürümleri yükler; npm run dev Vite geliştirme sunucusunu, npm run build production çıktısını ve npm run preview yerel önizlemeyi başlatır. Oxlint, React hook kuralları ve component export yapısı için statik kontrol sağlar.

```
npm ci
npm run dev
npm run lint
npm run build
npm run preview
```

9.2. Vercel Dağıtımı

MyVid AI statik Vite SPA olarak Vercel üzerinde yayımlanmıştır. Vercel production sürecinde proje bağımlılıklarını kurar, Vite build komutunu çalıştırır ve dist klasöründeki statik varlıkları dağıtır. vercel.json içindeki rewrite, doğrudan annotation rotasına gidildiğinde sunucunun 404 döndürmesi yerine React uygulamasının açılmasını sağlar. Aynı yapı gerekli COOP/COEP başlıklarını tüm rotalara ekler.

```
{
  "rewrites": [
    { "source": "/(.*)", "destination": "/index.html" }
  ],
  "headers": [{
    "source": "/(.*)",
    "headers": [
      { "key": "Cross-Origin-Opener-Policy", "value": "same-origin" },
      { "key": "Cross-Origin-Embedder-Policy", "value": "require-corp" }
    ]
  }]
}
```

9.3. Çalışma Koşulları

- Güncel Chrome veya Edge gibi modern bir tarayıcı.
- IndexedDB, Canvas 2D, WebAssembly ve WebGL desteği.
- Video boyutuna uygun RAM ve tarayıcı depolama kotası.
- İlk MediaPipe model yüklemesi ve Gemini kullanımı için internet erişimi.
- Kaynak video codec'ini oynatabilen tarayıcı.

10. TEST VE DOĞRULAMA

Proje hem yerel geliştirme ortamında hem de yayımlanan site üzerinde çalışır durumdadır. Doğrulama; kullanıcı akışları, kaynak kodu içindeki event ve state bağlantıları, kalıcı veri katmanı, servis entegrasyonları ve dağıtım yapılandırması üzerinden değerlendirilmiştir. API anahtarı veya kişisel video gerektiren testler kullanıcı kontrollü özellik olarak ele alınmıştır.

Uygulama açılışı	Karşılama ekranı ve dashboard yönlendirmesi	Başarılı
Proje oluşturma	Video IndexedDB'ye, metadata Redux'a kaydedilir	Başarılı
Video oynatma	Play/pause, seek, hız ve zaman kontrolleri çalışır	Başarılı
Manuel annotation	Kutu çizilir, seçilir, taşınır ve etiketlenir	Başarılı
AI nesne tespiti	Geçerli karede nesne kutuları oluşturulur	Destekleniyor
Nesne takibi	Keyframe rotası ve ilerleme durumu üretilir	Destekleniyor
JSON çıktısı	Annotation dizisi indirilebilir dosyaya dönüşür	Başarılı
Video kesme	Takip aralığı FFmpeg ile çıktı haline gelir	Destekleniyor
Gemini analizi	Storyboard karelerinden Türkçe açıklama alınır	İsteğe bağlı
SPA dağıtımı	Alt rotalar Vercel rewrite ile açılır	Başarılı

Tablo 10.1. İşlevsel doğrulama matrisi

10.1. Bağımlılık ve Kod Kalitesi

package-lock sürüm 3 biçimindedir ve uygulamanın doğrudan/dolaylı bağımlılıklarını bütünlük değerleriyle sabitler. Gerçekleştirilen lockfile güvenlik taramasında bilinen güvenlik açığı raporlanmamıştır. Oxlint yapılandırmasında React hook kuralları hata düzeyinde kontrol edilir. Kodun servis, slice, feature ve UI katmanlarına ayrılması bakım ve yeni özellik geliştirme açısından olumlu bir yapı sunar.

10.2. Tarayıcı ve Format Değerlendirmesi

Tarayıcı	Güncel Chromium tabanlı tarayıcı	Canvas, WASM, WebGL ve codec desteği güçlü
Video	H.264/AAC MP4	Oynatma ve MP4 stream-copy için en uyumlu seçenek
WebM	VP8/VP9/Opus	Oynatma desteklenir; MP4 kesit çıktısında codec uyumu dikkate alınır
Büyük dosya	Yeterli RAM ve depolama	FFmpeg işlemi dosyayı tarayıcı belleğinde işleyebilir
Mobil	Modern mobil tarayıcı	Temel arayüz responsive; yoğun AI işlemi cihaz gücüne bağlıdır

Tablo 10.2. Tarayıcı ve format değerlendirme

10.3. Performans Yaklaşımı

- MediaPipe modeli ilk ihtiyaçta başlatılarak başlangıç deneyimi hafif tutulur.
- Video dosyası Redux yerine IndexedDB'de saklanır.
- Thumbnail maksimum 320 piksel genişliğe küçültülür.
- Gemini kareleri yarı çözünürlükte ve JPEG kalitesinde hazırlanır.
- Takip adımı video süresine göre büyütülerek uzun videolarda işlem sayısı dengelenir.
- Konva node'ları playback sırasında doğrudan güncellenerek sürekli Redux render'ı azaltılır.
- FFmpeg çekirdeği yalnızca kesme özelliği kullanıldığında yüklenir.

11. SINIRLAMALAR VE GELECEK ÇALIŞMALAR

MyVid AI belirlenen kısa geliştirme döneminde hedeflenen ana kullanıcı akışlarını tamamlamıştır. Projenin mevcut sınırları, frontend tabanlı mimarinin doğal genişleme alanları olarak değerlendirilmiştir. Takip yaklaşımı sade ve anlaşılırdır; birbirine çok benzeyen nesnelerin yakınlaştığı karmaşık sahnelerde daha gelişmiş re-identification yöntemleri gelecekte doğruluğu artırabilir. Büyük ve yüksek çözünürlüklü videolarda işlem süresi cihaz kaynaklarına bağlıdır.

Tarayıcı origin'ine bağlı yerel depolama, gizlilik ve kurulum kolaylığı sağlarken cihazlar arası senkronizasyon sunmaz. İlerleyen sürümlerde isteğe bağlı kullanıcı hesabı, şifreli bulut yedekleme veya JSON içe aktarma eklenebilir. FFmpeg kesme katmanı codec farklarını otomatik algılayarak stream-copy ile yeniden kodlama arasında seçim yapacak biçimde genişletilebilir. Annotation veri şemasına sürüm bilgisi, görünülük aralığı ve içe aktarma doğrulaması eklemek veri taşınabilirliğini güçlendirebilir.

11.1. Önerilen Geliştirmeler

- Web Worker tabanlı MediaPipe yürütmesi ile uzun takip işlemlerinde arayüz akıcılığını artırma.
- Sınıf, IoU, hız ve görünüm özelliklerini birleştiren gelişmiş nesne eşleştirme.
- Annotation JSON içe aktarma ve şema sürümlenme.
- Codec algılama, otomatik çıktı formatı ve gerektiğinde yeniden kodlama.
- Klavye erişilebilirliği, ARIA etiketleri ve kapsamlı responsive testleri.

- Unit, integration ve uçtan uca test paketleri.
- İsteğe bağlı proje yedekleme ve cihazlar arası aktarım.
- Güncel Google GenAI SDK'sına geçiş ve API anahtar yönetiminin güçlendirilmesi.

Genel değerlendirme

Bu geliştirme alanları mevcut projenin çalışabilirliğini azaltan unsurlar değil; tamamlanmış MVP'nin daha geniş kullanıcı ve üretim senaryolarına taşınması için doğal bir yol haritasıdır.

12. SONUÇ

Bu bitirme projesinde tarayıcı tabanlı video inceleme, annotation ve yapay zeka işlevlerini tek bir kullanıcı deneyiminde birleştiren MyVid AI geliştirilmiştir. Uygulama; proje oluşturma, video saklama, thumbnail üretme, özel oynatma kontrolleri, manuel kutu çizme, AI ile nesne tespiti, keyframe tabanlı takip, timeline, video kesme, Gemini sahne analizi ve JSON dışı aktarma işlevlerini bütünleşik biçimde sunmaktadır.

React ve özellik odaklı klasör yapısı kullanıcı arayüzünü modüler hale getirirken Redux Toolkit paylaşılan durumu yönetmiştir. IndexedDB büyük video dosyaları için uygun yerel saklama katmanı sağlamış, göreceli koordinat modeli annotation'ların responsive ekranlarda korunmasına yardımcı olmuştur. MediaPipe ve FFmpeg.wasm gibi istemci tarafı teknolojiler, yoğun işlemlerin uygulama sunucusuna bağımlı olmadan gerçekleştirilmesini mümkün kılmıştır. Gemini entegrasyonu ise kullanıcının isteğine bağlı multimodal yorumlama katmanı eklemiştir.

Proje, Frontend Developer Programı kapsamında React mimarisi, durum yönetimi, tarayıcı depolaması, Canvas etkileşimi, WebAssembly, yapay zeka servis entegrasyonu, responsive tasarım ve statik dağıtım konularını gerçek bir ürün senaryosunda bir araya getirmiştir. Yerelde ve yayın ortamında çalışan MyVid AI, belirlenen proje hedeflerini karşılayan; geliştirilebilir, kullanıcı odaklı ve teknik kapsamı güçlü bir bireysel bitirme projesidir.

MyVid AI - Videolarınızı yalnızca izlemeyin; anlayın, işaretleyin ve önemli anları ortaya çıkarın.

13. KAYNAKÇA

- [1] React Team. *React Documentation: Quick Start and Managing State*. <https://react.dev/learn> (Eriřim: 16 Temmuz 2026).
- [2] Vite Team. *Vite Guide - Getting Started and Building for Production*. <https://vite.dev/guide/> ve <https://vite.dev/guide/build> (Eriřim: 16 Temmuz 2026).
- [3] Redux Team. *Redux Toolkit - Official Documentation*. <https://redux-toolkit.js.org/> (Eriřim: 16 Temmuz 2026).
- [4] React Router. *React Router Documentation*. <https://reactrouter.com/> (Eriřim: 16 Temmuz 2026).
- [5] Google AI Edge. *Object Detection Guide for Web*. https://developers.google.com/edge/mediapipe/solutions/vision/object_detector/web_js (Eriřim: 16 Temmuz 2026).
- [6] Google AI for Developers. *Gemini API - Image Understanding*. <https://ai.google.dev/gemini-api/docs/image-understanding> (Eriřim: 16 Temmuz 2026).
- [7] Google AI for Developers. *Gemini 3.5 Flash Model Documentation*. <https://ai.google.dev/gemini-api/docs/models/gemini-3.5-flash> (Eriřim: 16 Temmuz 2026).
- [8] ffmpeg.wasm Project. *ffmpeg.wasm Overview and Architecture*. <https://ffmpegwasm.netlify.app/docs/overview/> (Eriřim: 16 Temmuz 2026).
- [9] Konva Project. *React Konva Documentation*. <https://konvajs.org/docs/react/> (Eriřim: 16 Temmuz 2026).
- [10] Mozilla Developer Network. *IndexedDB API*. https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API (Eriřim: 16 Temmuz 2026).
- [11] Mozilla Developer Network. *HTMLVideoElement Web API*. <https://developer.mozilla.org/en-US/docs/Web/API/HTMLVideoElement> (Eriřim: 16 Temmuz 2026).
- [12] Jake Archibald. *idb - IndexedDB with Usability*. <https://github.com/jakearchibald/idb> (Eriřim: 16 Temmuz 2026).
- [13] Vercel. *Vite on Vercel*. <https://vercel.com/docs/frameworks/frontend/vite> (Eriřim: 16 Temmuz 2026).
- [14] Tailwind Labs. *Tailwind CSS Documentation*. <https://tailwindcss.com/docs> (Eriřim: 16 Temmuz 2026).
- [15] MyVid AI kaynak kodu, package.json, package-lock.json, README.md ve uygulama servisleri. Ünal Öztürk, Temmuz 2026.