

Eyewear Web Application - Final Technical Documentation

Version 2.0 - India Market Optimized | Dark Theme Implementation

Executive Summary

This document provides complete technical specifications for developing a modern eyewear e-commerce web application optimized for the Indian market. The application features a sophisticated dark theme design, virtual try-on capabilities, comprehensive eyewear catalog, doctor finder, eye health resources, and seamless e-commerce integration.

Target Market: India (Primary focus on metro cities: Delhi, Mumbai, Bangalore, Hyderabad, Chennai, Pune)

Technology Stack: Next.js 14+, React 18+, TypeScript, Tailwind CSS, Node.js, MongoDB/PostgreSQL

Development Timeline: 12-16 weeks across 4 phases

1. Design System - Dark Theme Color Palette

1.1 Color Palette (Verified & Optimized)

Based on dark mode best practices, we avoid pure black (#000000) and pure white (#FFFFFF) to reduce eye strain and improve readability^{[1] [2] [3]}.

Primary Colors

```
/* Background Layers - Dark Grays */
--bg-primary: #121212;          /* Main background (Material Design standard) */
--bg-secondary: #1E1E1E;        /* Cards, elevated surfaces */
--bg-tertiary: #2A2A2A;         /* Buttons, interactive elements */
--bg-hover: #353535;            /* Hover states */
--bg-active: #404040;           /* Active/pressed states */

/* Brand Colors - Desaturated for Dark Mode */
--brand-primary: #32B8C6;        /* Teal (desaturated from #00CED1) */
--brand-primary-hover: #3DD5E3;
--brand-primary-active: #28A5B1;
--brand-secondary: #E8A55C;      /* Gold/Amber (desaturated from #F59E0B) */
--brand-secondary-hover: #F0B572;
--brand-secondary-active: #D89447;

/* Text Colors - Muted Whites */
--text-primary: #E8E8E8;        /* Primary text (not pure white) */
```

```

--text-secondary: #B4B4B4;    /* Secondary text, labels */
--text-tertiary: #8A8A8A;     /* Disabled, placeholders */
--text-inverse: #1A1A1A;      /* Text on light backgrounds */

/* Semantic Colors */
--success: #4CAF50;           /* Green (slightly desaturated) */
--success-bg: rgba(76, 175, 80, 0.15);
--warning: #FFA726;           /* Orange (desaturated) */
--warning-bg: rgba(255, 167, 38, 0.15);
--error: #EF5350;             /* Red (slightly desaturated) */
--error-bg: rgba(239, 83, 80, 0.15);
--info: #42A5F5;              /* Blue (desaturated) */
--info-bg: rgba(66, 165, 245, 0.15);

/* Borders & Dividers */
--border-primary: rgba(232, 232, 232, 0.12);
--border-secondary: rgba(232, 232, 232, 0.08);
--border-hover: rgba(232, 232, 232, 0.20);

/* Overlay & Shadow */
--overlay: rgba(0, 0, 0, 0.7);
--shadow-sm: 0 2px 4px rgba(0, 0, 0, 0.3);
--shadow-md: 0 4px 8px rgba(0, 0, 0, 0.4);
--shadow-lg: 0 8px 16px rgba(0, 0, 0, 0.5);

```

Frame Color Highlights (Catalog Display)

```

/* Popular frame colors for product showcase */
--frame-tortoise: #8B5A3C;
--frame-black: #1F1F1F;
--frame-crystal: rgba(232, 232, 232, 0.25);
--frame-gold: #D4AF37;
--frame-silver: #C0C0C0;
--frame-blue: #4A90E2;
--frame-pink: #E06B9F;
--frame-green: #6FAF6B;

```

1.2 Typography System

```

/* Font Families */
--font-primary: 'Inter', -apple-system, BlinkMacSystemFont, 'Segoe UI', sans-serif;
--font-heading: 'Poppins', 'Inter', sans-serif;
--font-mono: 'Fira Code', 'Courier New', monospace;

/* Font Sizes (Responsive Scale) */
--text-xs: clamp(0.75rem, 0.7rem + 0.25vw, 0.875rem); /* 12-14px */
--text-sm: clamp(0.875rem, 0.825rem + 0.25vw, 1rem); /* 14-16px */
--text-base: clamp(1rem, 0.95rem + 0.25vw, 1.125rem); /* 16-18px */
--text-lg: clamp(1.125rem, 1.05rem + 0.375vw, 1.25rem); /* 18-20px */
--text-xl: clamp(1.25rem, 1.15rem + 0.5vw, 1.5rem); /* 20-24px */
--text-2xl: clamp(1.5rem, 1.35rem + 0.75vw, 2rem); /* 24-32px */
--text-3xl: clamp(2rem, 1.75rem + 1.25vw, 3rem); /* 32-48px */
--text-4xl: clamp(2.5rem, 2rem + 2.5vw, 4rem); /* 40-64px */

```

```

/* Font Weights */
--weight-light: 300;
--weight-normal: 400;
--weight-medium: 500;
--weight-semibold: 600;
--weight-bold: 700;

/* Line Heights */
--leading-tight: 1.2;
--leading-snug: 1.375;
--leading-normal: 1.5;
--leading-relaxed: 1.625;
--leading-loose: 2;

/* Letter Spacing */
--tracking-tight: -0.025em;
--tracking-normal: 0;
--tracking-wide: 0.025em;
--tracking-wider: 0.05em;

```

1.3 Spacing & Layout

```

/* Spacing Scale (8px base unit) */
--space-1: 0.25rem;    /* 4px */
--space-2: 0.5rem;     /* 8px */
--space-3: 0.75rem;    /* 12px */
--space-4: 1rem;       /* 16px */
--space-5: 1.25rem;    /* 20px */
--space-6: 1.5rem;     /* 24px */
--space-8: 2rem;       /* 32px */
--space-10: 2.5rem;    /* 40px */
--space-12: 3rem;      /* 48px */
--space-16: 4rem;      /* 64px */
--space-20: 5rem;      /* 80px */
--space-24: 6rem;      /* 96px */

/* Border Radius */
--radius-sm: 0.25rem;  /* 4px */
--radius-md: 0.5rem;   /* 8px */
--radius-lg: 0.75rem;  /* 12px */
--radius-xl: 1rem;     /* 16px */
--radius-2xl: 1.5rem;  /* 24px */
--radius-full: 9999px; /* Fully rounded */

/* Container Widths */
--container-sm: 640px;
--container-md: 768px;
--container-lg: 1024px;
--container-xl: 1280px;
--container-2xl: 1536px;

/* Z-Index Layers */
--z-base: 0;
--z-dropdown: 1000;

```

```
--z-sticky: 1020;  
--z-fixed: 1030;  
--z-modal-backdrop: 1040;  
--z-modal: 1050;  
--z-popover: 1060;  
--z-tooltip: 1070;
```

1.4 Animation & Transitions

```
/* Duration */  
--duration-fast: 150ms;  
--duration-normal: 250ms;  
--duration-slow: 350ms;  
--duration-slower: 500ms;  
  
/* Easing Functions */  
--ease-in: cubic-bezier(0.4, 0, 1, 1);  
--ease-out: cubic-bezier(0, 0, 0.2, 1);  
--ease-in-out: cubic-bezier(0.4, 0, 0.2, 1);  
--ease-bounce: cubic-bezier(0.68, -0.55, 0.265, 1.55);  
--ease-smooth: cubic-bezier(0.16, 1, 0.3, 1);  
  
/* Common Transitions */  
--transition-all: all var(--duration-normal) var(--ease-in-out);  
--transition-colors: color var(--duration-fast) var(--ease-out),  
                      background-color var(--duration-fast) var(--ease-out),  
                      border-color var(--duration-fast) var(--ease-out);  
--transition-transform: transform var(--duration-normal) var(--ease-smooth);  
--transition-opacity: opacity var(--duration-normal) var(--ease-in-out);
```

2. UI Component Specifications

2.1 Buttons

```
// Primary Button (CTA - Teal)  
<button className="btn-primary">  
  Shop Now  
</button>  
  
/* CSS */  
.btn-primary {  
  background: var(--brand-primary);  
  color: var(--text-inverse);  
  padding: var(--space-3) var(--space-6);  
  border-radius: var(--radius-lg);  
  font-weight: var(--weight-semibold);  
  font-size: var(--text-base);  
  border: none;  
  cursor: pointer;  
  transition: var(--transition-colors);  
  box-shadow: var(--shadow-sm);
```

```

}

.btn-primary:hover {
  background: var(--brand-primary-hover);
  box-shadow: var(--shadow-md);
  transform: translateY(-2px);
}

.btn-primary:active {
  background: var(--brand-primary-active);
  transform: translateY(0);
}

// Secondary Button (Outline)
.btn-secondary {
  background: transparent;
  color: var(--text-primary);
  padding: var(--space-3) var(--space-6);
  border: 2px solid var(--border-primary);
  border-radius: var(--radius-lg);
  font-weight: var(--weight-medium);
  transition: var(--transition-all);
}

.btn-secondary:hover {
  border-color: var(--brand-primary);
  color: var(--brand-primary);
  background: rgba(50, 184, 198, 0.08);
}

// Icon Button
.btn-icon {
  background: var(--bg-tertiary);
  color: var(--text-primary);
  padding: var(--space-3);
  border-radius: var(--radius-md);
  border: none;
  cursor: pointer;
  display: inline-flex;
  align-items: center;
  justify-content: center;
  transition: var(--transition-colors);
}

.btn-icon:hover {
  background: var(--bg-hover);
}

```

2.2 Cards (Product/Content)

```

<div>
  <div>
    <img>
      &lt;button className="btn-wishlist"&gt;♥&lt;/button&gt;
    </div>

```

```

<div>
  <span>Best Seller</span>
  <h3>Tortoise Shell Aviator</h3>
  <p>
    <span>₹2,499</span>
    <span>₹3,999</span>
    <span>38% OFF</span>
  </p>
  <div>
    <span></span>
    <span></span>
    <span></span>
  </div>
  <button className="btn-primary btn-full">Try Virtual</button>
</div>
</div>

```

```

/* CSS */
.card-product {
  background: var(--bg-secondary);
  border-radius: var(--radius-xl);
  overflow: hidden;
  border: 1px solid var(--border-primary);
  transition: var(--transition-all);
  cursor: pointer;
}

.card-product:hover {
  border-color: var(--brand-primary);
  box-shadow: 0 8px 24px rgba(50, 184, 198, 0.15);
  transform: translateY(-4px);
}

.card-image {
  position: relative;
  aspect-ratio: 4/3;
  background: var(--bg-tertiary);
  overflow: hidden;
}

.card-image img {
  width: 100%;
  height: 100%;
  object-fit: cover;
  transition: transform var(--duration-slow) var(--ease-out);
}

.card-product:hover .card-image img {
  transform: scale(1.05);
}

.btn-wishlist {
  position: absolute;
  top: var(--space-4);
  right: var(--space-4);
  background: rgba(18, 18, 18, 0.7);
}

```

```
    backdrop-filter: blur(8px);
    border: none;
    border-radius: var(--radius-full);
    padding: var(--space-2);
    cursor: pointer;
    font-size: var(--text-lg);
}

.card-content {
    padding: var(--space-6);
}

.card-badge {
    display: inline-block;
    background: var(--brand-secondary);
    color: var(--text-inverse);
    padding: var(--space-1) var(--space-3);
    border-radius: var(--radius-sm);
    font-size: var(--text-xs);
    font-weight: var(--weight-semibold);
    margin-bottom: var(--space-3);
}

.card-title {
    font-size: var(--text-lg);
    font-weight: var(--weight-semibold);
    color: var(--text-primary);
    margin: var(--space-2) 0;
}

.card-price {
    display: flex;
    align-items: center;
    gap: var(--space-2);
    margin: var(--space-4) 0;
}

.price-current {
    font-size: var(--text-xl);
    font-weight: var(--weight-bold);
    color: var(--brand-primary);
}

.price-original {
    font-size: var(--text-sm);
    color: var(--text-tertiary);
    text-decoration: line-through;
}

.price-discount {
    font-size: var(--text-xs);
    color: var(--success);
    font-weight: var(--weight-semibold);
}

.card-colors {
```

```

display: flex;
gap: var(--space-2);
margin: var(--space-4) 0;
}

.color-swatch {
width: 24px;
height: 24px;
border-radius: var(--radius-full);
border: 2px solid var(--border-primary);
cursor: pointer;
transition: var(--transition-transform);
}

.color-swatch:hover {
transform: scale(1.15);
border-color: var(--brand-primary);
}

```

2.3 Navigation Bar

```

<!-- nav className="navbar">
  <div>
    <div>
      <img>
      <span>Eyewear India</span>
    </div>

    <div>
      <input type="search" placeholder="Search glasses, sunglasses..." />
      <button className="search-btn"></button>
    </div>

    <div>
      <button className="nav-btn">
        <span></span>
        <span>Stores</span>
      </button>
      <button className="nav-btn">
        <span>♥</span>
        <span>Wishlist</span>
        <span>3</span>
      </button>
      <button className="nav-btn">
        <span></span>
        <span>Cart</span>
        <span>2</span>
      </button>
      <button className="nav-btn-avatar">
        <img>
      </button>
    </div>
  </div>

</div>

```



```

    <a href="#eyeglasses">Eyeglasses</a>
    <a href="#sunglasses">Sunglasses</a>
    <a href="#computer-glasses">Computer Glasses</a>
    <a href="#face-shape">Face Shape Guide</a>
    <a href="#doctors">Find Doctor</a>
    <a href="#deals">Deals <img alt="heart icon" data-bbox="385 121 400 136"/></a>
  </div>
</nav>

/* CSS */
.navbar {
  background: var(--bg-secondary);
  border-bottom: 1px solid var(--border-primary);
  position: sticky;
  top: 0;
  z-index: var(--z-sticky);
  backdrop-filter: blur(12px);
  background: rgba(30, 30, 30, 0.95);
}

.navbar-container {
  max-width: var(--container-2xl);
  margin: 0 auto;
  padding: var(--space-4) var(--space-6);
  display: flex;
  align-items: center;
  justify-content: space-between;
  gap: var(--space-8);
}

.navbar-logo {
  display: flex;
  align-items: center;
  gap: var(--space-3);
  font-size: var(--text-xl);
  font-weight: var(--weight-bold);
  color: var(--brand-primary);
}

.navbar-logo img {
  height: 32px;
}

.navbar-search {
  flex: 1;
  max-width: 600px;
  position: relative;
}

.navbar-search input {
  width: 100%;
  padding: var(--space-3) var(--space-12) var(--space-3) var(--space-4);
  background: var(--bg-tertiary);
  border: 1px solid var(--border-primary);
  border-radius: var(--radius-lg);
  color: var(--text-primary);
}

```

```
    font-size: var(--text-base);
    transition: var(--transition-colors);
}

.navbar-search input:focus {
    outline: none;
    border-color: var(--brand-primary);
    background: var(--bg-hover);
}

.search-btn {
    position: absolute;
    right: var(--space-2);
    top: 50%;
    transform: translateY(-50%);
    background: var(--brand-primary);
    border: none;
    border-radius: var(--radius-md);
    padding: var(--space-2) var(--space-4);
    cursor: pointer;
}

.navbar-actions {
    display: flex;
    align-items: center;
    gap: var(--space-4);
}

.nav-btn {
    position: relative;
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: var(--space-1);
    padding: var(--space-2) var(--space-4);
    background: transparent;
    border: none;
    color: var(--text-secondary);
    cursor: pointer;
    transition: var(--transition-colors);
    border-radius: var(--radius-md);
}

.nav-btn:hover {
    color: var(--brand-primary);
    background: var(--bg-tertiary);
}

.nav-btn .badge {
    position: absolute;
    top: 0;
    right: 0;
    background: var(--error);
    color: var(--text-primary);
    font-size: var(--text-xs);
    font-weight: var(--weight-bold);
```

```

padding: 2px 6px;
border-radius: var(--radius-full);
min-width: 18px;
text-align: center;
}

.navbar-menu {
max-width: var(--container-2xl);
margin: 0 auto;
padding: var(--space-3) var(--space-6);
display: flex;
align-items: center;
gap: var(--space-8);
overflow-x: auto;
}

.navbar-menu a {
color: var(--text-secondary);
text-decoration: none;
font-size: var(--text-base);
font-weight: var(--weight-medium);
padding: var(--space-2) var(--space-3);
border-radius: var(--radius-md);
transition: var(--transition-colors);
white-space: nowrap;
}

.navbar-menu a:hover {
color: var(--brand-primary);
background: var(--bg-tertiary);
}

```

2.4 Form Inputs

```

/* Input Field */
.form-group {
margin-bottom: var(--space-6);
}

.form-label {
display: block;
font-size: var(--text-sm);
font-weight: var(--weight-medium);
color: var(--text-secondary);
margin-bottom: var(--space-2);
}

.form-input {
width: 100%;
padding: var(--space-4);
background: var(--bg-tertiary);
border: 1px solid var(--border-primary);
border-radius: var(--radius-lg);
color: var(--text-primary);
font-size: var(--text-base);
}

```

```

    transition: var(--transition-colors);
}

.form-input:focus {
    outline: none;
    border-color: var(--brand-primary);
    box-shadow: 0 0 0 3px rgba(50, 184, 198, 0.15);
}

.form-input::placeholder {
    color: var(--text-tertiary);
}

/* Select Dropdown */
.form-select {
    width: 100%;
    padding: var(--space-4);
    background: var(--bg-tertiary);
    border: 1px solid var(--border-primary);
    border-radius: var(--radius-lg);
    color: var(--text-primary);
    font-size: var(--text-base);
    cursor: pointer;
    appearance: none;
    background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' wic
    background-repeat: no-repeat;
    background-position: right var(--space-4) center;
    padding-right: var(--space-12);
}

/* Checkbox */
.form-checkbox {
    display: flex;
    align-items: center;
    gap: var(--space-3);
    cursor: pointer;
}

.form-checkbox input[type="checkbox"] {
    width: 20px;
    height: 20px;
    border: 2px solid var(--border-primary);
    border-radius: var(--radius-sm);
    appearance: none;
    cursor: pointer;
    position: relative;
    background: var(--bg-tertiary);
    transition: var(--transition-colors);
}

.form-checkbox input[type="checkbox"]:checked {
    background: var(--brand-primary);
    border-color: var(--brand-primary);
}

.form-checkbox input[type="checkbox"]:checked::after {

```

```

    content: '✓';
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    color: var(--text-inverse);
    font-size: 14px;
  }

```

2.5 Modal/Overlay

```

<div>
  <div>
    <div>
      <h2>Virtual Try-On</h2>
      &lt;button className="modal-close"&gt;×&lt;/button>
    </div>
    <div>

      </div>
      <div>
        &lt;button className="btn-secondary"&gt;Cancel&lt;/button>
        &lt;button className="btn-primary"&gt;Add to Cart&lt;/button>
      </div>
    </div>
  </div>
</div>

/* CSS */
.modal-overlay {
  position: fixed;
  inset: 0;
  background: var(--overlay);
  backdrop-filter: blur(4px);
  display: flex;
  align-items: center;
  justify-content: center;
  z-index: var(--z-modal-backdrop);
  padding: var(--space-4);
  animation: fadeIn var(--duration-normal) var(--ease-out);
}

.modal {
  background: var(--bg-secondary);
  border: 1px solid var(--border-primary);
  border-radius: var(--radius-2xl);
  max-width: 600px;
  width: 100%;
  max-height: 90vh;
  overflow: hidden;
  display: flex;
  flex-direction: column;
  z-index: var(--z-modal);
  animation: slideUp var(--duration-normal) var(--ease-smooth);
  box-shadow: var(--shadow-lg);
}

```

```
.modal-header {
  display: flex;
  align-items: center;
  justify-content: space-between;
  padding: var(--space-6);
  border-bottom: 1px solid var(--border-primary);
}
```

```
.modal-title {
  font-size: var(--text-2xl);
  font-weight: var(--weight-semibold);
  color: var(--text-primary);
}
```

```
.modal-close {
  background: transparent;
  border: none;
  color: var(--text-secondary);
  font-size: var(--text-2xl);
  cursor: pointer;
  padding: var(--space-2);
  border-radius: var(--radius-md);
  transition: var(--transition-colors);
}
```

```
.modal-close:hover {
  background: var(--bg-hover);
  color: var(--text-primary);
}
```

```
.modal-body {
  padding: var(--space-6);
  overflow-y: auto;
  flex: 1;
}
```

```
.modal-footer {
  display: flex;
  gap: var(--space-4);
  padding: var(--space-6);
  border-top: 1px solid var(--border-primary);
  justify-content: flex-end;
}
```

```
@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}
```

```
@keyframes slideUp {
  from {
    opacity: 0;
    transform: translateY(20px);
  }
  to {
```

```
    opacity: 1;
    transform: translateY(0);
  }
}
```

3. Icon System

3.1 Icon Library

Use **Lucide React** icons (MIT licensed, optimized for React):

```
npm install lucide-react
```

3.2 Essential Icons

```
import {
  // Navigation
  Home, Search, Menu, X, ChevronDown, ChevronRight,

  // E-commerce
  ShoppingCart, Heart, Eye, Package, CreditCard,

  // User
  User, UserCircle, LogIn, LogOut, Settings,

  // Product
  Glasses, SunMedium, Monitor, Shield, Star,

  // Communication
  Phone, Mail, MessageCircle, MapPin,

  // Actions
  Plus, Minus, Edit, Trash2, Share2, Download,

  // Status
  CheckCircle, AlertCircle, Info, XCircle,

  // Social
  Facebook, Instagram, Twitter, Youtube
} from 'lucide-react';

// Usage
<ShoppingCart size={20} color="var(--text-primary)" strokeWidth={2} />
```

3.3 Custom SVG Icons

```
// Virtual Try-On Icon
export const VirtualTryOnIcon = () => {
  <svg width="24" height="24" viewBox="0 0 24 24" fill="none" stroke="currentColor">
    <path d="M3 12a9 9 0 0 0 9 9m9-9a9 9 0 0 1-9 9m0-18v18" stroke-width="2" stroke-linecap="round" />
    <ellipse cx="12" cy="9" rx="8" ry="4" stroke-width="2" />
    <path d="M6 12a6 6 0 0 0 6 3 6 6 0 0 0 6-3" stroke-width="2" stroke-linecap="round" />
  </svg>
};

// Face Shape Icon
export const FaceShapeIcon = () => {
  <svg width="24" height="24" viewBox="0 0 24 24" fill="none" stroke="currentColor">
    <ellipse cx="12" cy="13" rx="7" ry="9" stroke-width="2" />
    <circle cx="9" cy="11" r="1.5" fill="currentColor" />
    <circle cx="15" cy="11" r="1.5" fill="currentColor" />
    <path d="M9 16 Q12 18 15 16" stroke-width="2" stroke-linecap="round" />
  </svg>
};

// Eye Doctor Icon
export const EyeDoctorIcon = () => {
  <svg width="24" height="24" viewBox="0 0 24 24" fill="none" stroke="currentColor">
    <path d="M2 12s3-7 10-7 10 7 3 7-10 7-10-7z" stroke-width="2" />
    <circle cx="12" cy="12" r="3" stroke-width="2" />
    <path d="M12 5v2m0 10v2m5 12h3m18 0h-2" stroke-width="2" stroke-linecap="round" />
  </svg>
};
```

4. Page-by-Page Specifications

4.1 Home Page

Layout Sections:

1. Hero Section

- Full-viewport height with gradient overlay
- Main CTA: "Find Your Perfect Glasses"
- Virtual try-on quick access button
- Background: Dynamic image carousel of diverse Indian models wearing eyewear

```
<section className="hero">
  <div>
    <span>India's #1 Eyewear Platform </span>
    <h1>
      Discover Eyewear That<br>
      <span>Defines Your Style</span>
    </h1>
    <p>
```



```

    2000+ Frames | Virtual Try-On | Free Home Delivery | Expert Guidance
  </p>
  <div>
    <button className="btn-primary btn-lg">
      Start Virtual Try-On
    </button>
    <button className="btn-secondary btn-lg">
      Browse Collection
    </button>
  </div>
  <div>
    <div>
      <span>5M+</span>
      <span>Happy Customers</span>
    </div>
    <div>
      <span>2000+</span>
      <span>Frame Styles</span>
    </div>
    <div>
      <span>4.8★</span>
      <span>Average Rating</span>
    </div>
  </div>
</div>
</section>

/* CSS */
.hero {
  min-height: 100vh;
  display: flex;
  align-items: center;
  justify-content: center;
  background: linear-gradient(135deg,
    rgba(18, 18, 18, 0.95) 0%,
    rgba(30, 30, 30, 0.9) 100%),
    url('/hero-bg.jpg');
  background-size: cover;
  background-position: center;
  position: relative;
}

.hero-content {
  max-width: var(--container-lg);
  text-align: center;
  padding: var(--space-8);
}

.hero-badge {
  display: inline-block;
  background: rgba(50, 184, 198, 0.15);
  border: 1px solid rgba(50, 184, 198, 0.3);
  color: var(--brand-primary);
  padding: var(--space-2) var(--space-4);
  border-radius: var(--radius-full);
  font-size: var(--text-sm);
}

```

```
    font-weight: var(--weight-semibold);
    margin-bottom: var(--space-6);
}

.hero-title {
    font-size: var(--text-4xl);
    font-weight: var(--weight-bold);
    color: var(--text-primary);
    line-height: var(--leading-tight);
    margin-bottom: var(--space-6);
}

.gradient-text {
    background: linear-gradient(135deg, var(--brand-primary), var(--brand-secondary));
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
    background-clip: text;
}

.hero-subtitle {
    font-size: var(--text-xl);
    color: var(--text-secondary);
    max-width: 600px;
    margin: 0 auto var(--space-8);
}

.hero-actions {
    display: flex;
    gap: var(--space-4);
    justify-content: center;
    flex-wrap: wrap;
    margin-bottom: var(--space-12);
}

.hero-stats {
    display: flex;
    gap: var(--space-12);
    justify-content: center;
    flex-wrap: wrap;
    margin-top: var(--space-16);
}

.stat {
    display: flex;
    flex-direction: column;
    align-items: center;
    gap: var(--space-2);
}

.stat-number {
    font-size: var(--text-3xl);
    font-weight: var(--weight-bold);
    color: var(--brand-primary);
}

.stat-label {
```

```
font-size: var(--text-sm);  
color: var(--text-secondary);  
}
```

2. Featured Categories

- Grid: 3 columns (desktop), 2 (tablet), 1 (mobile)
- Categories: Eyeglasses, Sunglasses, Computer Glasses
- Hover effect: Scale + glow

3. Best Sellers

- Horizontal scroll on mobile
- Grid on desktop (4 columns)
- Product cards with virtual try-on badge

4. Face Shape Guide CTA

- Interactive banner
- "Find frames for your face shape"
- Link to face shape quiz

5. Trending Colors

- Visual color palette showcase
- Tortoiseshell, Clear, Blue, Gold highlighted
- Quick filter to catalog

6. Eye Health Section

- Informational cards
- Blue light protection info
- UV protection importance
- Link to doctor finder

7. Customer Testimonials

- Carousel with photos
- Star ratings
- Real customer names and cities

8. Brand Logos

- Trusted brands: Ray-Ban, Lenskart, Titan Eye+, etc.
- Horizontal scroll

4.2 Product Catalog Page

Features:

- Left sidebar: Filters (collapsible on mobile)
- Main area: Product grid (4-3-2-1 responsive columns)
- Top bar: Sort options, view toggle (grid/list)

Filters:

```
<!-- Filters -->
<aside className="filters">
  <div>
    <h3>Gender</h3>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Men</span>
    </label>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Women</span>
    </label>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Unisex</span>
    </label>
  </div>

  <div>
    <h3>Frame Type</h3>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Full Rim</span>
    </label>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Semi-Rimless</span>
    </label>
    <label className="filter-option">
      <input type="checkbox" />
      <span>Rimless</span>
    </label>
  </div>

  <div>
    <h3>Price Range</h3>
    <input type="range" min="0" max="10000" className="price-slider" />
    <div>
      <span>₹0</span>
      <span>₹10,000</span>
    </div>
  </div>

  <div>
    <h3>Frame Material</h3>
    <label className="filter-option">
```

```

        <input type="checkbox" />
        <span>Acetate</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Titanium</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Stainless Steel</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>TR90</span>
    </label>
</div>

<div>
    <h3>Frame Color</h3>
    <div>
        <button className="color-filter" style="background: #8B5A3C" title="Tortoiseshell">Tortoiseshell</button>
        <button className="color-filter" style="background: #1F1F1F" title="Black">Black</button>
        <button className="color-filter" style="background: rgba(232,232,232,0.3)" title="Transparent">Transparent</button>
        <button className="color-filter" style="background: #D4AF37" title="Gold">Gold</button>
        <button className="color-filter" style="background: #C0C0C0" title="Silver">Silver</button>
        <button className="color-filter" style="background: #4A90E2" title="Blue">Blue</button>
    </div>
</div>

<div>
    <h3>Face Shape</h3>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Oval</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Round</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Square</span>
    </label>
    <label className="filter-option">
        <input type="checkbox" />
        <span>Heart</span>
    </label>
</div>

    <button className="btn-secondary btn-full">Reset Filters</button>
</aside>

```

Sort Options:

- Relevance

- Price: Low to High
- Price: High to Low
- Newest First
- Most Popular
- Highest Rated

4.3 Product Detail Page

Layout:

```
<div>
  <div>
    <div>
      <img>
        &lt;button className="gallery-zoom"&gt;&lt;/button&gt;
      </div>
      <div>
        <img>
        <img>
        <img>
        <img>
      </div>
      &lt;button className="btn-virtual"&gt;
        &lt;VirtualTryOnIcon /&gt;
        Try Virtual
      &lt;/button&gt;
    </div>

    <div>
      <div>
        <span>Lenskart</span>
        <span>
          ★★★★★☆ 4.5 (234 reviews)
        </span>
      </div>

      <h1>
        Tortoise Shell Aviator Premium Eyeglasses
      </h1>

      <p>
        Classic aviator design with premium acetate tortoiseshell pattern.
        Lightweight titanium temples for all-day comfort. UV400 protection included.
      </p>

      <div>
        <span>₹2,499</span>
        <span>₹3,999</span>
        <span>38% OFF</span>
      </div>

      <div>
        <div>
```

```

    ☐ EMI starting from ₹833/month
  </div>
</div>
<div>
  ☐ Free Home Delivery
</div>
<div>
  ✓ 14 Days Return Policy
</div>
</div>

<div>
  <div>
    <label className="option-label">Frame Color</label>
    <div>
      <button className="color-option active" style="background: #8B5A3C"></button>
      <button className="color-option" style="background: #1F1F1F"></button>
      <button className="color-option" style="background: #4A90E2"></button>
    </div>
  </div>

  <div>
    <label className="option-label">Lens Type</label>
    <select className="form-select">
      <option>Single Vision</option>
      <option>Bifocal</option>
      <option>Progressive</option>
      <option>Blue Light Blocking</option>
    </select>
  </div>

  <div>
    <label className="option-label">Lens Coating</label>
    <div>
      <label className="form-checkbox">
        <input type="checkbox" checked="" />
        <span>Anti-Reflective (₹500)</span>
      </label>
      <label className="form-checkbox">
        <input type="checkbox" />
        <span>Scratch Resistant (₹300)</span>
      </label>
      <label className="form-checkbox">
        <input type="checkbox" />
        <span>UV Protection (₹400)</span>
      </label>
    </div>
  </div>
</div>

<div>
  <button className="btn-primary btn-lg btn-full">
    Add to Cart - ₹2,499
  </button>
  <button className="btn-secondary btn-lg btn-full">
    Buy Now
  </button>

```

```
<button className="btn-icon btn-lg">
  <Heart />
</button>
</div>

<div>
  <h3>Specifications</h3>
  <div>
    <span>Frame Material</span>
    <span>Acetate</span>
  </div>
  <div>
    <span>Frame Width</span>
    <span>140mm</span>
  </div>
  <div>
    <span>Lens Width</span>
    <span>52mm</span>
  </div>
  <div>
    <span>Bridge Width</span>
    <span>18mm</span>
  </div>
  <div>
    <span>Temple Length</span>
    <span>145mm</span>
  </div>
  <div>
    <span>Weight</span>
    <span>28g</span>
  </div>
  <div>
    <span>Suitable Face Shape</span>
    <span>Oval, Round, Square</span>
  </div>
</div>
</div>

<section className="product-reviews">
  <h2>Customer Reviews</h2>

</section>

<section className="related-products">
  <h2>Similar Products</h2>

</section>
```


4.4 Virtual Try-On Page

Requirements:

- Camera access permission
- Face detection using TensorFlow.js or MediaPipe
- Real-time frame overlay
- Capture photo functionality
- Share to social media

```
<div>
  <div>
    <video id="camera-feed" autoPlay></video>
    <canvas id="overlay-canvas"></canvas>

    <div>
      <button className="btn-icon" title="Switch Camera">
        </button>
      <button className="btn-primary btn-capture">
        Capture Photo
      </button>
      <button className="btn-icon" title="Share">
        <Share2 />
      </button>
    </div>
  </div>

  <div>
    <h3>Select Frame</h3>
    <div>

    </div>

    <div>
      <div>
        <label>Frame Size</label>
        <input type="range" min="0.8" max="1.2" step="0.05" value="1" />
      </div>
      <div>
        <label>Position</label>
        <div>
          <button className="btn-icon">↑</button>
          <button className="btn-icon">↓</button>
          <button className="btn-icon">←</button>
          <button className="btn-icon">→</button>
        </div>
      </div>
    </div>

    <button className="btn-primary btn-full">
      Add to Cart
    </button>
```

```
</div>
</div>
```

4.5 Face Shape Guide Page

Interactive Quiz:

```
<div>
  &lt;section className="guide-hero"&gt;
    <h1>Find Your Perfect Frame</h1>
    <p>
      Discover which glasses suit your face shape best
    </p>
  &lt;/section&gt;

  &lt;section className="face-shapes"&gt;
    <div>
      <img>
      <h3>Oval Face</h3>
      <p>
        Balanced proportions with slightly wider cheekbones
      </p>
      <div>
        <h4>Best Frames:</h4>
        <ul>
          <li>Square frames</li>
          <li>Rectangular frames</li>
          <li>Geometric designs</li>
        </ul>
      </div>
      &lt;button className="btn-secondary"&gt;View Frames&lt;/button&gt;
    </div>

    &lt;/section&gt;

    &lt;section className="face-quiz"&gt;
      <h2>Take Our Face Shape Quiz</h2>
      <div>
        <div>
          <h3>Step 1: Upload Photo or Use Camera</h3>
          <div>
            &lt;button className="btn-primary"&gt;Upload Photo&lt;/button&gt;
            <span>or</span>
            &lt;button className="btn-secondary"&gt;Use Camera&lt;/button&gt;
          </div>
        </div>

        <div>
          <h3>Step 2: Answer Quick Questions</h3>
        </div>

        <div>
          <h3>Step 3: Get Your Results</h3>
        </div>
      </div>
    </section>
  </div>
```

```

    </div>
  </div>
</section>
</div>

```

4.6 Doctor Finder Page

```

<div>
  <section className="finder-search">
    <h1>Find Eye Care Professionals Near You</h1>

    <div>
      <input
        type="text"
        placeholder="Enter your city or pincode"
        className="form-input"
      />
      <select className="form-select">
        <option>Ophthalmologist</option>
        <option>Optometrist</option>
        <option>Optician</option>
      </select>
      <button className="btn-primary">Search</button>
    </div>

    <div>
      <button className="filter-chip">Available Today</button>
      <button className="filter-chip">Online Consultation</button>
      <button className="filter-chip">Top Rated</button>
      <button className="filter-chip">Accepts Insurance</button>
    </div>
  </section>

  <section className="finder-results">
    <div>

    </div>

    <div>
      <div>
        <img>
        <div>
          <h3>Dr. Rajesh Kumar</h3>
          <p>Ophthalmologist (MD)</p>
          <p>15 years experience</p>
          <div>
            ★★★★★ 4.9 (156 reviews)
          </div>
          <p>
            📍 Fortis Hospital, Delhi
          </p>
          <p>
            Consultation Fee: ₹1,200
          </p>

```

```

        </div>
        <div>
            <button className="btn-primary">Book Appointment</button>
            <button className="btn-secondary">View Profile</button>
        </div>
    </div>

    </div>
    </div>
    </div>

```

4.7 Eye Health Resources Page

```

<div>
    <section className="health-hero">
        <h1>Eye Health & Care Guide</h1>
        <p>
            Expert advice for maintaining healthy vision
        </p>
    </section>

    <section className="health-categories">
        <div>
            <div></div>
            <h3>Vision Problems</h3>
            <ul>
                <li><a href="#myopia">Myopia (Nearsightedness)</a></li>
                <li><a href="#hyperopia">Hyperopia (Farsightedness)</a></li>
                <li><a href="#astigmatism">Astigmatism</a></li>
                <li><a href="#presbyopia">Presbyopia</a></li>
            </ul>
        </div>

        <div>
            <div></div>
            <h3>Digital Eye Strain</h3>
            <ul>
                <li><a href="#blue-light">Blue Light Protection</a></li>
                <li><a href="#screen-time">Managing Screen Time</a></li>
                <li><a href="#exercises">Eye Exercises</a></li>
            </ul>
        </div>

    </section>

    <section className="health-articles">
        <h2>Latest Articles</h2>
        <div>
            <article className="article-card">
                <img>
                <div>
                    <span>Prevention</span>
                    <h3>

```

```
        10 Tips to Protect Your Eyes from Blue Light
    </h3>
    <p>
        Learn how to reduce digital eye strain with these simple tips...
    </p>
    <a href="#">Read More →</a>
</div>
<!-- Article Content -->
</div>
</div>
</div>
```

5. Development Phases (12-16 Weeks)

Phase 1: Foundation & Setup (Weeks 1-2)

Week 1: Project Initialization

- [x] Set up Next.js 14+ project with TypeScript
- [x] Configure Tailwind CSS with custom dark theme
- [x] Set up Git repository and branching strategy
- [x] Configure ESLint, Prettier, and Husky
- [x] Set up project folder structure
- [x] Install core dependencies (Lucide icons, date-fns, clsx)
- [x] Create design tokens (CSS variables)
- [x] Set up Storybook for component documentation

Week 2: Core Components

- [x] Build Button variants (primary, secondary, icon)
- [x] Build Form components (Input, Select, Checkbox, Radio)
- [x] Build Card components (Product, Content)
- [x] Build Navigation (Header, Footer)
- [x] Build Modal/Dialog component
- [x] Build Toast notification system
- [x] Test responsive behavior (mobile, tablet, desktop)

Deliverables:

- Working component library
- Style guide documentation
- Responsive grid system

Phase 2: Product Catalog & E-Commerce (Weeks 3-6)

Week 3: Database & API Setup

- [x] Set up MongoDB/PostgreSQL database
- [x] Design database schema (Products, Users, Orders, Reviews)
- [x] Create API routes for CRUD operations
- [x] Implement product filtering logic
- [x] Set up image storage (Cloudinary/AWS S3)
- [x] Create seed data (200+ products)

Week 4: Product Pages

- [x] Home page with hero section
- [x] Featured categories section
- [x] Best sellers carousel
- [x] Product catalog page with filters
- [x] Product detail page
- [x] Related products section
- [x] Quick view modal

Week 5: Search & Filtering

- [x] Implement search functionality
- [x] Advanced filters (price, color, material, face shape)
- [x] Sort options (price, popularity, rating)
- [x] Filter persistence (URL params)
- [x] Mobile filter drawer
- [x] Loading states and skeletons

Week 6: Shopping Cart & Wishlist

- [x] Shopping cart page
- [x] Cart sidebar/drawer
- [x] Add/remove items
- [x] Update quantities
- [x] Cart persistence (localStorage)
- [x] Wishlist functionality
- [x] Price calculations (discounts, taxes)

Deliverables:

- Fully functional product catalog

- Working cart and wishlist
- Search and filter system

Phase 3: Advanced Features (Weeks 7-10)

Week 7: Virtual Try-On (Basic)

- [x] Camera access implementation
- [x] Face detection using MediaPipe/TensorFlow.js
- [x] Frame overlay rendering
- [x] Photo capture functionality
- [x] Frame selection interface
- [x] Basic adjustment controls

Week 8: Virtual Try-On (Advanced)

- [x] Improve face landmark detection accuracy
- [x] Add frame size adjustment
- [x] Add position adjustment (rotation, tilt)
- [x] Multiple camera support (front/back)
- [x] Share to social media
- [x] Save try-on photos

Week 9: Face Shape Guide

- [x] Face shape information pages
- [x] Interactive face shape quiz
- [x] Photo upload for analysis
- [x] AI-based face shape detection (optional)
- [x] Recommended frames per face shape
- [x] Filter catalog by face shape

Week 10: Doctor Finder

- [x] Google Maps integration
- [x] Search doctors by location
- [x] Doctor profile pages
- [x] Filter by specialty
- [x] Display consultation fees
- [x] Booking appointment functionality (basic)
- [x] Doctor reviews and ratings

Deliverables:

- Working virtual try-on feature
- Face shape guide and quiz
- Doctor finder with maps

Phase 4: User Experience & Polish (Weeks 11-14)**Week 11: User Authentication**

- [x] Sign up / Login pages
- [x] Social login (Google, Facebook)
- [x] Email verification
- [x] Password reset flow
- [x] User profile page
- [x] Order history
- [x] Saved addresses

Week 12: Checkout Process

- [x] Multi-step checkout flow
- [x] Shipping address form
- [x] Payment gateway integration (Razorpay/Stripe)
- [x] Order summary
- [x] Coupon code application
- [x] Order confirmation page
- [x] Email notifications

Week 13: Eye Health & Content

- [x] Eye health resource pages
- [x] Article/blog system
- [x] Vision problems information
- [x] Blue light protection guide
- [x] Eye care tips
- [x] FAQ section

Week 14: Performance & SEO

- [x] Image optimization (Next.js Image)
- [x] Lazy loading
- [x] Code splitting

- [x] Meta tags for all pages
- [x] Open Graph tags
- [x] Sitemap generation
- [x] robots.txt
- [x] [Schema.org](https://schema.org) structured data

Deliverables:

- Complete user authentication
- Full checkout flow
- Eye health content
- SEO optimization

Phase 5: Testing & Deployment (Weeks 15-16)

Week 15: Testing

- [x] Unit tests (Jest)
- [x] Component tests (React Testing Library)
- [x] E2E tests (Playwright/Cypress)
- [x] Cross-browser testing
- [x] Mobile device testing
- [x] Accessibility audit (WCAG 2.1 AA)
- [x] Performance testing (Lighthouse)
- [x] Security audit

Week 16: Deployment & Launch

- [x] Set up production environment
- [x] Configure CI/CD pipeline
- [x] Deploy to Vercel/AWS
- [x] Set up monitoring (Sentry, Google Analytics)
- [x] Final QA testing
- [x] Load testing
- [x] Soft launch with limited traffic
- [x] Full production launch

Deliverables:

- Fully tested application
- Production deployment

- Monitoring and analytics

6. Key Functional Requirements

6.1 Virtual Try-On System

Technical Implementation:

```
// Face Detection using MediaPipe
import { FaceMesh } from '@mediapipe/face_mesh';
import { Camera } from '@mediapipe/camera_utils';

class VirtualTryOn {
  private faceMesh: FaceMesh;
  private camera: Camera;
  private canvas: HTMLCanvasElement;
  private ctx: CanvasRenderingContext2D;

  constructor(videoElement: HTMLVideoElement, canvasElement: HTMLCanvasElement) {
    this.canvas = canvasElement;
    this.ctx = canvasElement.getContext('2d')!;

    // Initialize MediaPipe Face Mesh
    this.faceMesh = new FaceMesh({
      locateFile: (file) => `https://cdn.jsdelivr.net/npm/@mediapipe/face_mesh/${file}`
    });

    this.faceMesh.setOptions({
      maxNumFaces: 1,
      refineLandmarks: true,
      minDetectionConfidence: 0.5,
      minTrackingConfidence: 0.5
    });

    this.faceMesh.onResults(this.onResults.bind(this));

    // Initialize camera
    this.camera = new Camera(videoElement, {
      onFrame: async () => {
        await this.faceMesh.send({ image: videoElement });
      },
      width: 1280,
      height: 720
    });
  }

  onResults(results: any) {
    // Clear canvas
    this.ctx.clearRect(0, 0, this.canvas.width, this.canvas.height);

    // Draw video frame
    this.ctx.drawImage(results.image, 0, 0, this.canvas.width, this.canvas.height);
  }
}
```

```

if (results.multiFaceLandmarks && results.multiFaceLandmarks.length > 0) {
  const landmarks = results.multiFaceLandmarks[0];

  // Get key points for glasses placement
  const leftEye = landmarks[^33]; // Left eye outer corner
  const rightEye = landmarks[^263]; // Right eye outer corner
  const noseBridge = landmarks[^168]; // Nose bridge

  // Calculate glasses dimensions and position
  const eyeDistance = Math.sqrt(
    Math.pow((rightEye.x - leftEye.x) * this.canvas.width, 2) +
    Math.pow((rightEye.y - leftEye.y) * this.canvas.height, 2)
  );

  const glassesWidth = eyeDistance * 2.2; // Glasses wider than eye distance
  const glassesHeight = glassesWidth * 0.4; // Aspect ratio

  // Calculate center position
  const centerX = (leftEye.x + rightEye.x) * this.canvas.width / 2;
  const centerY = noseBridge.y * this.canvas.height;

  // Calculate rotation angle
  const angle = Math.atan2(
    rightEye.y - leftEye.y,
    rightEye.x - leftEye.x
  );

  // Draw glasses frame
  this.drawGlasses(centerX, centerY, glassesWidth, glassesHeight, angle);
}

drawGlasses(x: number, y: number, width: number, height: number, angle: number) {
  this.ctx.save();
  this.ctx.translate(x, y);
  this.ctx.rotate(angle);

  // Load and draw selected frame image
  const frameImage = new Image();
  frameImage.src = this.selectedFrameUrl;

  this.ctx.drawImage(
    frameImage,
    -width / 2,
    -height / 2,
    width,
    height
  );

  this.ctx.restore();
}

start() {
  this.camera.start();
}

```

```

    stop() {
        this.camera.stop();
    }

    capturePhoto(): string {
        return this.canvas.toDataURL('image/png');
    }

    changeFrame(frameUrl: string) {
        this.selectedFrameUrl = frameUrl;
    }
}

// Usage
const video = document.getElementById('camera-feed') as HTMLVideoElement;
const canvas = document.getElementById('overlay-canvas') as HTMLCanvasElement;
const tryOn = new VirtualTryOn(video, canvas);

tryOn.start();

```

Performance Optimization:

- Use Web Workers for face detection processing
- Implement frame throttling (process every 2-3 frames)
- Use WebGL for faster rendering
- Cache frame images
- Lazy load MediaPipe models

6.2 Advanced Search & Filtering

Algorithm:

```

interface FilterOptions {
    gender?: string[];
    frameType?: string[];
    frameMaterial?: string[];
    frameColor?: string[];
    faceShape?: string[];
    priceRange?: [number, number];
    brands?: string[];
    features?: string[];
}

interface SortOptions {
    field: 'price' | 'popularity' | 'rating' | 'newest';
    order: 'asc' | 'desc';
}

class ProductSearch {
    async searchProducts(
        query: string,
        filters: FilterOptions,

```

```

    sort: SortOptions,
    page: number = 1,
    limit: number = 24
) {
  // Build MongoDB aggregation pipeline
  const pipeline = [];

  // Text search
  if (query) {
    pipeline.push({
      $search: {
        index: 'products',
        text: {
          query: query,
          path: ['name', 'description', 'brand', 'tags'],
          fuzzy: { maxEdits: 2 }
        }
      }
    });
  }

  // Apply filters
  const matchStage: any = {};

  if (filters.gender?.length) {
    matchStage.gender = { $in: filters.gender };
  }

  if (filters.frameType?.length) {
    matchStage.frameType = { $in: filters.frameType };
  }

  if (filters.frameMaterial?.length) {
    matchStage.material = { $in: filters.frameMaterial };
  }

  if (filters.frameColor?.length) {
    matchStage.color = { $in: filters.frameColor };
  }

  if (filters.faceShape?.length) {
    matchStage.suitableFaceShapes = { $in: filters.faceShape };
  }

  if (filters.priceRange) {
    matchStage.price = {
      $gte: filters.priceRange[0],
      $lte: filters.priceRange[1]
    };
  }

  if (filters.brands?.length) {
    matchStage.brand = { $in: filters.brands };
  }

  if (filters.features?.length) {

```

```

    matchStage.features = { $all: filters.features };
  }

  if (Object.keys(matchStage).length > 0) {
    pipeline.push({ $match: matchStage });
  }

  // Sorting
  const sortStage: any = {};

  switch (sort.field) {
    case 'price':
      sortStage.price = sort.order === 'asc' ? 1 : -1;
      break;
    case 'popularity':
      sortStage.salesCount = sort.order === 'asc' ? 1 : -1;
      break;
    case 'rating':
      sortStage.averageRating = sort.order === 'asc' ? 1 : -1;
      break;
    case 'newest':
      sortStage.createdAt = sort.order === 'asc' ? 1 : -1;
      break;
  }

  pipeline.push({ $sort: sortStage });

  // Pagination
  const skip = (page - 1) * limit;
  pipeline.push({ $skip: skip });
  pipeline.push({ $limit: limit });

  // Add metadata
  pipeline.push({
    $facet: {
      products: [],
      totalCount: [{ $count: 'count' }]
    }
  });

  const results = await db.collection('products').aggregate(pipeline).toArray();

  return {
    products: results[0].products,
    total: results[0].totalCount[0]?.count || 0,
    page,
    limit,
    totalPages: Math.ceil((results[0].totalCount[0]?.count || 0) / limit)
  };
}

// Get filter options with counts
async getFilterOptions(query: string = '') {
  const pipeline = [];

  if (query) {

```

```

    pipeline.push({
      $search: {
        index: 'products',
        text: { query, path: ['name', 'description', 'brand'] }
      }
    });
  }

  pipeline.push({
    $facet: {
      genders: [
        { $group: { _id: '$gender', count: { $sum: 1 } } },
        { $sort: { count: -1 } }
      ],
      frameTypes: [
        { $group: { _id: '$frameType', count: { $sum: 1 } } },
        { $sort: { count: -1 } }
      ],
      materials: [
        { $unwind: '$material' },
        { $group: { _id: '$material', count: { $sum: 1 } } },
        { $sort: { count: -1 } }
      ],
      colors: [
        { $unwind: '$color' },
        { $group: { _id: '$color', count: { $sum: 1 } } },
        { $sort: { count: -1 } }
      ],
      brands: [
        { $group: { _id: '$brand', count: { $sum: 1 } } },
        { $sort: { count: -1 } }
      ],
      priceRange: [
        {
          $group: {
            _id: null,
            minPrice: { $min: '$price' },
            maxPrice: { $max: '$price' }
          }
        }
      ]
    }
  });

  const results = await db.collection('products').aggregate(pipeline).toArray();
  return results[0];
}

```

6.3 Recommendation Engine

```
class RecommendationEngine {
  // Collaborative filtering
  async getPersonalizedRecommendations(userId: string, limit: number = 10) {
    // Get user's purchase history and browsing data
    const userHistory = await db.collection('user_interactions').find({
      userId,
      type: { $in: ['purchase', 'view', 'wishlist'] }
    }).toArray();

    const productIds = userHistory.map(h => h.productId);

    // Find similar users
    const similarUsers = await db.collection('user_interactions').aggregate([
      {
        $match: {
          productId: { $in: productIds },
          userId: { $ne: userId }
        }
      },
      {
        $group: {
          _id: '$userId',
          commonProducts: { $sum: 1 }
        }
      },
      {
        $sort: { commonProducts: -1 }
      },
      {
        $limit: 50
      }
    ]).toArray();

    const similarUserIds = similarUsers.map(u => u._id);

    // Get products liked by similar users
    const recommendations = await db.collection('user_interactions').aggregate([
      {
        $match: {
          userId: { $in: similarUserIds },
          productId: { $nin: productIds },
          type: { $in: ['purchase', 'wishlist'] }
        }
      },
      {
        $group: {
          _id: '$productId',
          score: { $sum: 1 }
        }
      },
      {
        $sort: { score: -1 }
      },
      {

```



```

    $limit: limit
  },
  {
    $lookup: {
      from: 'products',
      localField: '_id',
      foreignField: '_id',
      as: 'product'
    }
  },
  {
    $unwind: '$product'
  }
].toArray();

return recommendations.map(r => r.product);
}

// Content-based filtering
async getSimilarProducts(productId: string, limit: number = 6) {
  const product = await db.collection('products').findOne({ _id: productId });

  if (!product) return [];

  // Find similar products based on attributes
  const similar = await db.collection('products').aggregate([
    {
      $match: {
        _id: { $ne: productId },
        $or: [
          { frameType: product.frameType },
          { material: { $in: product.material } },
          { color: { $in: product.color } },
          { brand: product.brand },
          { suitableFaceShapes: { $in: product.suitableFaceShapes } }
        ]
      }
    },
    {
      $addFields: {
        similarityScore: {
          $add: [
            { $cond: [{ $eq: ['$frameType', product.frameType] }, 3, 0] },
            { $size: { $setIntersection: ['$material', product.material] } },
            { $size: { $setIntersection: ['$color', product.color] } },
            { $cond: [{ $eq: ['$brand', product.brand] }, 2, 0] },
            { $size: { $setIntersection: ['$suitableFaceShapes', product.suitableFaceShapes] } }
          ]
        }
      }
    },
    {
      $sort: { similarityScore: -1, averageRating: -1 }
    },
    {
      $limit: limit
    }
  ]);
}

```

```

    }
  ]).toArray();

  return similar;
}

// Face shape based recommendations
async getRecommendationsForFaceShape(faceShape: string, filters: FilterOptions = {}) {
  const matchStage: any = {
    suitableFaceShapes: faceShape
  };

  // Apply additional filters
  if (filters.gender?.length) {
    matchStage.gender = { $in: filters.gender };
  }

  if (filters.priceRange) {
    matchStage.price = {
      $gte: filters.priceRange[^0],
      $lte: filters.priceRange[^1]
    };
  }

  const products = await db.collection('products').find(matchStage)
    .sort({ averageRating: -1, salesCount: -1 })
    .limit(24)
    .toArray();

  return products;
}
}

```

6.4 Payment Integration (Razorpay)

```

import Razorpay from 'razorpay';

const razorpay = new Razorpay({
  key_id: process.env.RAZORPAY_KEY_ID!,
  key_secret: process.env.RAZORPAY_KEY_SECRET!
});

class PaymentService {
  async createOrder(amount: number, currency: string = 'INR', orderId: string) {
    const options = {
      amount: amount * 100, // Convert to paise
      currency,
      receipt: orderId,
      notes: {
        order_id: orderId
      }
    };

    const order = await razorpay.orders.create(options);
    return order;
  }
}

```

```

}

async verifyPayment(
  razorpayOrderId: string,
  razorpayPaymentId: string,
  razorpaySignature: string
): Promise<boolean> {
  const crypto = require('crypto');

  const body = razorpayOrderId + '|' + razorpayPaymentId;

  const expectedSignature = crypto
    .createHmac('sha256', process.env.RAZORPAY_KEY_SECRET!)
    .update(body.toString())
    .digest('hex');

  return expectedSignature === razorpaySignature;
}

async capturePayment(paymentId: string, amount: number) {
  const payment = await razorpay.payments.capture(
    paymentId,
    amount * 100, // Convert to paise
    'INR'
  );

  return payment;
}

async refundPayment(paymentId: string, amount?: number) {
  const refundOptions: any = {
    payment_id: paymentId
  };

  if (amount) {
    refundOptions.amount = amount * 100;
  }

  const refund = await razorpay.payments.refund(refundOptions);
  return refund;
}

// Frontend integration
const initializeRazorpay = () => {
  return new Promise((resolve) => {
    const script = document.createElement('script');
    script.src = 'https://checkout.razorpay.com/v1/checkout.js';
    script.onload = () => resolve(true);
    script.onerror = () => resolve(false);
    document.body.appendChild(script);
  });
};

const processPayment = async (order: any) => {
  const res = await initializeRazorpay();

```

```

if (!res) {
  alert('Razorpay SDK failed to load');
  return;
}

const options = {
  key: process.env.NEXT_PUBLIC_RAZORPAY_KEY_ID,
  amount: order.amount,
  currency: order.currency,
  name: 'EyeWear India',
  description: 'Eyewear Purchase',
  order_id: order.id,
  handler: async function (response: any) {
    // Verify payment on backend
    const verifyResponse = await fetch('/api/payment/verify', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({
        orderId: response.razorpay_order_id,
        paymentId: response.razorpay_payment_id,
        signature: response.razorpay_signature
      })
    });

    if (verifyResponse.ok) {
      // Payment successful
      window.location.href = '/order-success';
    } else {
      alert('Payment verification failed');
    }
  },
  prefill: {
    name: user.name,
    email: user.email,
    contact: user.phone
  },
  theme: {
    color: '#32B8C6'
  }
};

const paymentObject = new (window as any).Razorpay(options);
paymentObject.open();
};

```

7. API Endpoints Documentation

7.1 Products API

GET /api/products

Query params:

- q: string (search query)
- gender: string[]
- frameType: string[]
- material: string[]
- color: string[]
- faceShape: string[]
- minPrice: number
- maxPrice: number
- brand: string[]
- sortBy: 'price' | 'popularity' | 'rating' | 'newest'
- sortOrder: 'asc' | 'desc'
- page: number
- limit: number

Response:

```
{
  products: Product[],
  total: number,
  page: number,
  limit: number,
  totalPages: number
}
```

GET /api/products/:id

Response: Product

GET /api/products/:id/similar

Response: Product[]

POST /api/products (Admin only)

Body: ProductInput

Response: Product

PUT /api/products/:id (Admin only)

Body: Partial<ProductInput>;

Response: Product

DELETE /api/products/:id (Admin only)

Response: { success: boolean }

7.2 Cart API

GET /api/cart

Response: Cart

POST /api/cart/items

Body: { productId: string, quantity: number, options: object }

Response: Cart

PUT /api/cart/items/:itemId

```
Body: { quantity: number }  
Response: Cart  
  
DELETE /api/cart/items/:itemId  
Response: Cart  
  
DELETE /api/cart  
Response: { success: boolean }
```

7.3 Orders API

```
GET /api/orders  
Query params:  
  - status: string  
  - page: number  
  - limit: number  
Response: Order[]  
  
GET /api/orders/:id  
Response: Order  
  
POST /api/orders  
Body: {  
  items: CartItem[],  
  shippingAddress: Address,  
  billingAddress: Address,  
  paymentMethod: string  
}  
Response: Order  
  
PUT /api/orders/:id/cancel  
Response: Order  
  
GET /api/orders/:id/invoice  
Response: PDF file
```

7.4 Users API

```
POST /api/auth/register  
Body: { email, password, name }  
Response: { user: User, token: string }  
  
POST /api/auth/login  
Body: { email, password }  
Response: { user: User, token: string }  
  
POST /api/auth/logout  
Response: { success: boolean }  
  
GET /api/users/profile  
Response: User  
  
PUT /api/users/profile
```

```
Body: Partial<User>;
Response: User

POST /api/users/addresses
Body: Address
Response: Address

PUT /api/users/addresses/:id
Body: Partial<Address>;
Response: Address

DELETE /api/users/addresses/:id
Response: { success: boolean }
```

7.5 Reviews API

```
GET /api/products/:productId/reviews
Query params:
  - page: number
  - limit: number
  - sortBy: 'recent' | 'helpful'
Response: Review[]

POST /api/products/:productId/reviews
Body: { rating: number, title: string, content: string, images: string[] }
Response: Review

PUT /api/reviews/:id
Body: Partial<Review>;
Response: Review

DELETE /api/reviews/:id
Response: { success: boolean }

POST /api/reviews/:id/helpful
Response: Review
```

7.6 Doctors API

```
GET /api/doctors
Query params:
  - city: string
  - pincode: string
  - specialty: string
  - availability: string
  - lat: number
  - lng: number
  - radius: number
Response: Doctor[]

GET /api/doctors/:id
Response: Doctor
```

```
POST /api/doctors/:id/appointments
Body: { date: string, time: string, reason: string }
Response: Appointment
```

8. Security & Performance

8.1 Security Measures

```
// Rate limiting
import rateLimit from 'express-rate-limit';

const limiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 100, // Limit each IP to 100 requests per windowMs
  message: 'Too many requests from this IP'
});

app.use('/api/', limiter);

// CSRF protection
import csrf from 'csurf';
const csrfProtection = csrf({ cookie: true });

// Input validation
import { z } from 'zod';

const ProductSchema = z.object({
  name: z.string().min(3).max(100),
  description: z.string().min(10).max(2000),
  price: z.number().positive(),
  brand: z.string(),
  images: z.array(z.string().url()).min(1),
  // ... more fields
});

// SQL injection prevention (use parameterized queries)
// XSS prevention (sanitize user inputs)
import DOMPurify from 'isomorphic-dompurify';
const sanitized = DOMPurify.sanitize(userInput);

// Authentication with JWT
import jwt from 'jsonwebtoken';

const generateToken = (userId: string) => {
  return jwt.sign({ userId }, process.env.JWT_SECRET!, {
    expiresIn: '7d'
  });
};

const verifyToken = (token: string) => {
  return jwt.verify(token, process.env.JWT_SECRET!);
};
```



```
// Password hashing
import bcrypt from 'bcryptjs';

const hashPassword = async (password: string) => {
  return bcrypt.hash(password, 10);
};

const comparePassword = async (password: string, hash: string) => {
  return bcrypt.compare(password, hash);
};
```

8.2 Performance Optimization

```
// Image optimization with Next.js Image
import Image from 'next/image';

<Image
  src="/product.jpg"
  alt="Product"
  width={500}
  height={400}
  quality={80}
  placeholder="blur"
  blurDataURL="data:image/jpeg;base64,..."
  loading="lazy"
/>

// Code splitting
import dynamic from 'next/dynamic';

const VirtualTryOn = dynamic(() => import('@components/VirtualTryOn'), {
  loading: () => <LoadingSpinner />,
  ssr: false
});

// API response caching
import { Redis } from 'ioredis';
const redis = new Redis(process.env.REDIS_URL);

const getCachedData = async (key: string, fetchFn: () => Promise<any>) => {
  const cached = await redis.get(key);

  if (cached) {
    return JSON.parse(cached);
  }

  const data = await fetchFn();
  await redis.set(key, JSON.stringify(data), 'EX', 3600); // Cache for 1 hour

  return data;
};

// Database indexing
// MongoDB indexes
db.products.createIndex({ name: 'text', description: 'text', brand: 'text' });
```

```

db.products.createIndex({ gender: 1, frameType: 1, price: 1 });
db.products.createIndex({ brand: 1, averageRating: -1 });

// Lazy loading images
const [ref, inView] = useInView({
  triggerOnce: true,
  threshold: 0.1
});

return (
  <div>
    {inView & & <Image src={src} alt={alt} />}
  </div>
);

```

9. Accessibility Compliance

```

// WCAG 2.1 AA compliance checklist

// 1. Keyboard navigation
<button
  onClick={handleClick}
  onKeyDown={(e) => {
    if (e.key === 'Enter' || e.key === ' ') {
      handleClick();
    }
  }}
  tabIndex={0}
>
  Click me
</button>

// 2. ARIA labels
<button aria-label="Add to cart">
  <ShoppingCart />
</button>

<nav aria-label="Main navigation">
  {/* Navigation items */}
</nav>

// 3. Focus management
import { useEffect, useRef } from 'react';

const Modal = ({ isOpen }: { isOpen: boolean }) => {
  const firstFocusableRef = useRef<HTMLButtonElement>(null);

  useEffect(() => {
    if (isOpen & & firstFocusableRef.current) {
      firstFocusableRef.current.focus();
    }
  }, [isOpen]);

  return (

```

```

    <div>
      <button ref={firstFocusableRef}>Close</button>
      { /* Modal content */ }
    </div>
  );
};

// 4. Color contrast (minimum 4.5:1 for normal text)
// Already implemented in design system

// 5. Alternative text for images


// 6. Form labels
<label htmlFor="email">Email Address</label>
<input id="email" type="email" required aria-required="true" />

// 7. Error messages
<input
  aria-invalid={hasError}
  aria-describedby={hasError ? 'email-error' : undefined}
/>
{hasError & & (
  <span>
    Please enter a valid email address
  </span>
)}

// 8. Skip to main content
<a href="#main-content">
  Skip to main content
</a>

<main id="main-content">
  { /* Page content */ }
</main>

/* CSS */
.skip-link {
  position: absolute;
  top: -40px;
  left: 0;
  background: var(--brand-primary);
  color: var(--text-inverse);
  padding: var(--space-2) var(--space-4);
  z-index: 100;
}

.skip-link:focus {
  top: 0;
}

```

10. Analytics & Monitoring

```
// Google Analytics 4
import { initializeApp } from 'firebase/app';
import { getAnalytics, logEvent } from 'firebase/analytics';

const firebaseConfig = {
  apiKey: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
  // ... other config
};

const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);

// Track events
export const trackEvent = (eventName: string, params?: object) => {
  logEvent(analytics, eventName, params);
};

// Usage
trackEvent('product_view', {
  product_id: product.id,
  product_name: product.name,
  price: product.price
});

trackEvent('add_to_cart', {
  product_id: product.id,
  quantity: 1
});

// Error monitoring with Sentry
import * as Sentry from '@sentry/nextjs';

Sentry.init({
  dsn: process.env.NEXT_PUBLIC_SENTRY_DSN,
  environment: process.env.NODE_ENV,
  tracesSampleRate: 1.0
});

// Performance monitoring
import { performance } from 'perf_hooks';

const measurePerformance = (name: string) => {
  performance.mark(`${name}-start`);

  return () => {
    performance.mark(`${name}-end`);
    performance.measure(name, `${name}-start`, `${name}-end`);

    const measure = performance.getEntriesByName(name)[0];
    console.log(`${name} took ${measure.duration}ms`);
  };
};

// Usage
```

```
const endMeasure = measurePerformance('product-search');  
// ... perform search  
endMeasure();
```

11. Deployment Checklist

11.1 Pre-Deployment

- ☐ All tests passing (unit, integration, E2E)
- ☐ Performance audit (Lighthouse score > 90)
- ☐ Accessibility audit (WCAG 2.1 AA compliant)
- ☐ Security audit (no critical vulnerabilities)
- ☐ Cross-browser testing (Chrome, Firefox, Safari, Edge)
- ☐ Mobile device testing (iOS, Android)
- ☐ SEO audit (meta tags, sitemap, robots.txt)
- ☐ Error handling and logging configured
- ☐ Environment variables set correctly
- ☐ Database migrations ready
- ☐ CDN configured for static assets
- ☐ SSL certificate installed
- ☐ Backup strategy implemented

11.2 Post-Deployment

- ☐ Monitor error rates (Sentry)
- ☐ Monitor performance metrics
- ☐ Monitor server resources (CPU, memory, disk)
- ☐ Check analytics tracking working
- ☐ Verify payment gateway integration
- ☐ Test email notifications
- ☐ Verify database backups
- ☐ Check CDN cache behavior
- ☐ Monitor API response times
- ☐ Review user feedback

12. Maintenance & Support

12.1 Regular Tasks

Daily:

- Monitor error logs
- Check server health
- Review customer support tickets

Weekly:

- Update product inventory
- Review analytics reports
- Check website performance
- Update content (blogs, offers)

Monthly:

- Security updates
- Database optimization
- Content backup verification
- User feedback analysis
- A/B testing review

Quarterly:

- Major feature updates
- Design improvements
- Dependency updates
- Performance optimization

12.2 Scaling Strategy

Phase 1 (0-10K users/day):

- Single server deployment
- Managed database service
- CDN for static assets

Phase 2 (10K-100K users/day):

- Load balancer
- Multiple application servers
- Database read replicas

- Redis caching layer

Phase 3 (100K+ users/day):

- Microservices architecture
- Kubernetes orchestration
- Database sharding
- Advanced caching strategies
- Edge computing

13. Conclusion

This comprehensive technical documentation provides all specifications needed to build a world-class eyewear e-commerce platform optimized for the Indian market. The dark theme design, combined with cutting-edge features like virtual try-on and AI-powered recommendations, positions the application as a market leader.

Key Success Factors:

1. ✓ User-centric dark theme design optimized for reduced eye strain
2. ✓ Sophisticated color palette following best practices
3. ✓ Complete component library with consistent styling
4. ✓ Advanced virtual try-on technology
5. ✓ Intelligent search and recommendation systems
6. ✓ Comprehensive eye health resources
7. ✓ Seamless payment and checkout experience
8. ✓ Phased development approach for predictable delivery
9. ✓ Security and performance optimization built-in
10. ✓ Scalability and maintenance considerations

Next Steps:

1. Assemble development team
2. Set up development environment
3. Begin Phase 1 (Foundation & Setup)
4. Weekly progress reviews
5. Iterative testing and feedback incorporation

For questions or clarifications, refer to specific sections or contact the technical architect.

Document Version: 2.0

Last Updated: November 26, 2025

Status: Final - Ready for Development

[\[4\]](#) [\[5\]](#) [\[6\]](#) [\[7\]](#) [\[8\]](#) [\[9\]](#) [\[10\]](#) [\[11\]](#) [\[12\]](#) [\[13\]](#) [\[14\]](#) [\[15\]](#) [\[16\]](#) [\[17\]](#) [\[18\]](#) [\[19\]](#) [\[20\]](#)

**

1. <https://app.uxcel.com/tutorials/12-principles-of-dark-mode-design-627>
2. <https://www.starterstory.com/sunglasses-business-web-designs>
3. <https://www.mendix.com/blog/agile-software-development-lifecycle-stages/>
4. <https://www.designstudiouiux.com/blog/dark-mode-ui-design-best-practices/>
5. <https://dribbble.com/tags/eyewear>
6. <https://www.marceldigital.com/blog/what-is-agile-web-development-everything-you-need-to-know>
7. <https://www.interaction-design.org/literature/article/ui-color-palette>
8. [https://www.behance.net/search/projects/eyewear web design](https://www.behance.net/search/projects/eyewear%20web%20design)
9. <https://thinksys.com/development/agile-software-development-life-cycle/>
10. <https://m2.material.io/design/color/the-color-system.html>
11. <https://dribbble.com/tags/glasses-website>
12. <https://iprogrammer.com/web-application-development-life-cycle-a-2024-brief-guide-for-dedicated-developers/>
13. <https://www.litmus.com/blog/the-ultimate-guide-to-dark-mode-for-email-marketers>
14. <https://themeforest.net/search/eyewear>
15. <https://atmos.style/blog/dark-mode-ui-best-practices>
16. <https://futuresoftindia.com/blogs/top-10-eyewear-ecommerce-websites.html>
17. <https://www.geeksforgeeks.org/software-engineering/agile-sdlc-software-development-life-cycle/>
18. <https://thisisglance.com/learning-centre/what-are-the-best-practices-for-dark-mode-colour-schemes>
19. <https://imatrix.com/blog/best-optician-website-designs/>
20. <https://www.brightwork.com/blog/5-stages-of-the-agile-system-development-life-cycle>