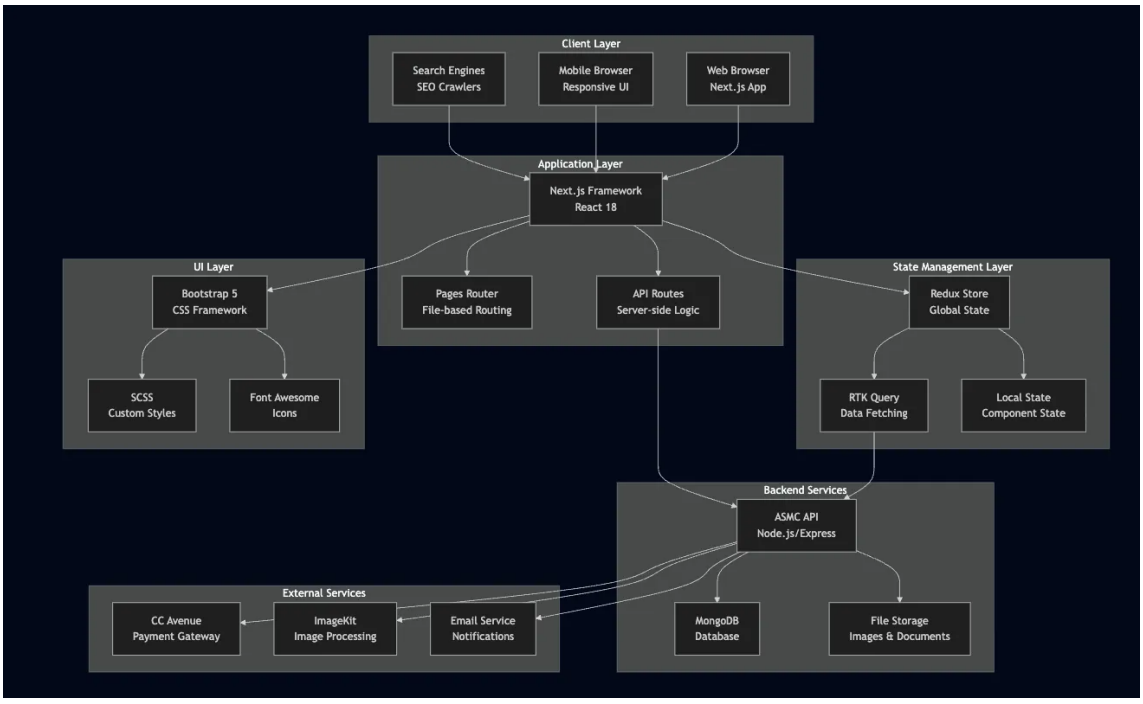
ASMC Next - Architecture Overview

Comprehensive architectural documentation for the ASMC Next.js frontend application, including system design, component architecture, data flow, and integration patterns.

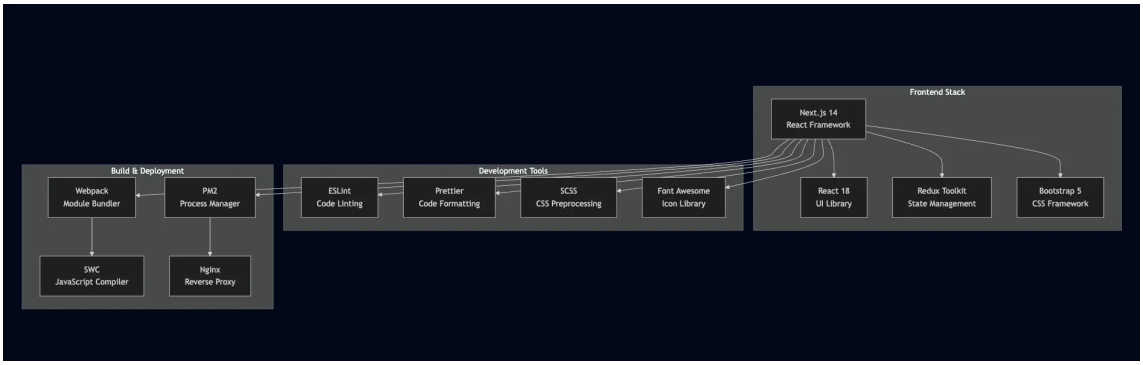
Table of Contents

System Architecture

High-Level Architecture

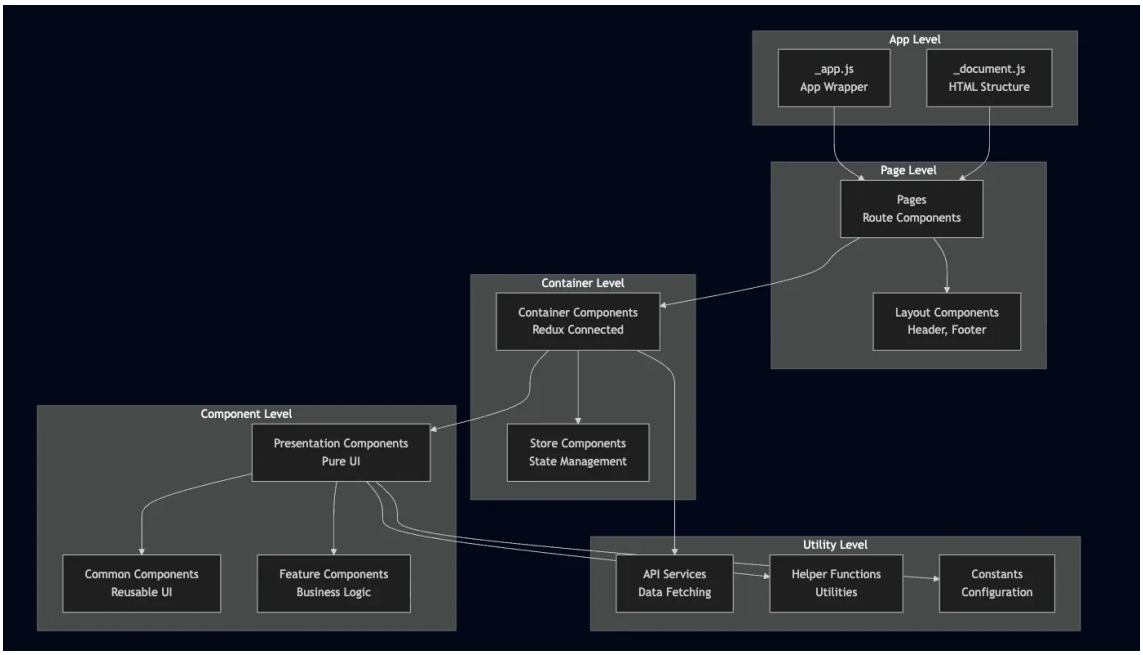


Technology Stack Architecture



Component Architecture

Component Hierarchy



Component Pattern

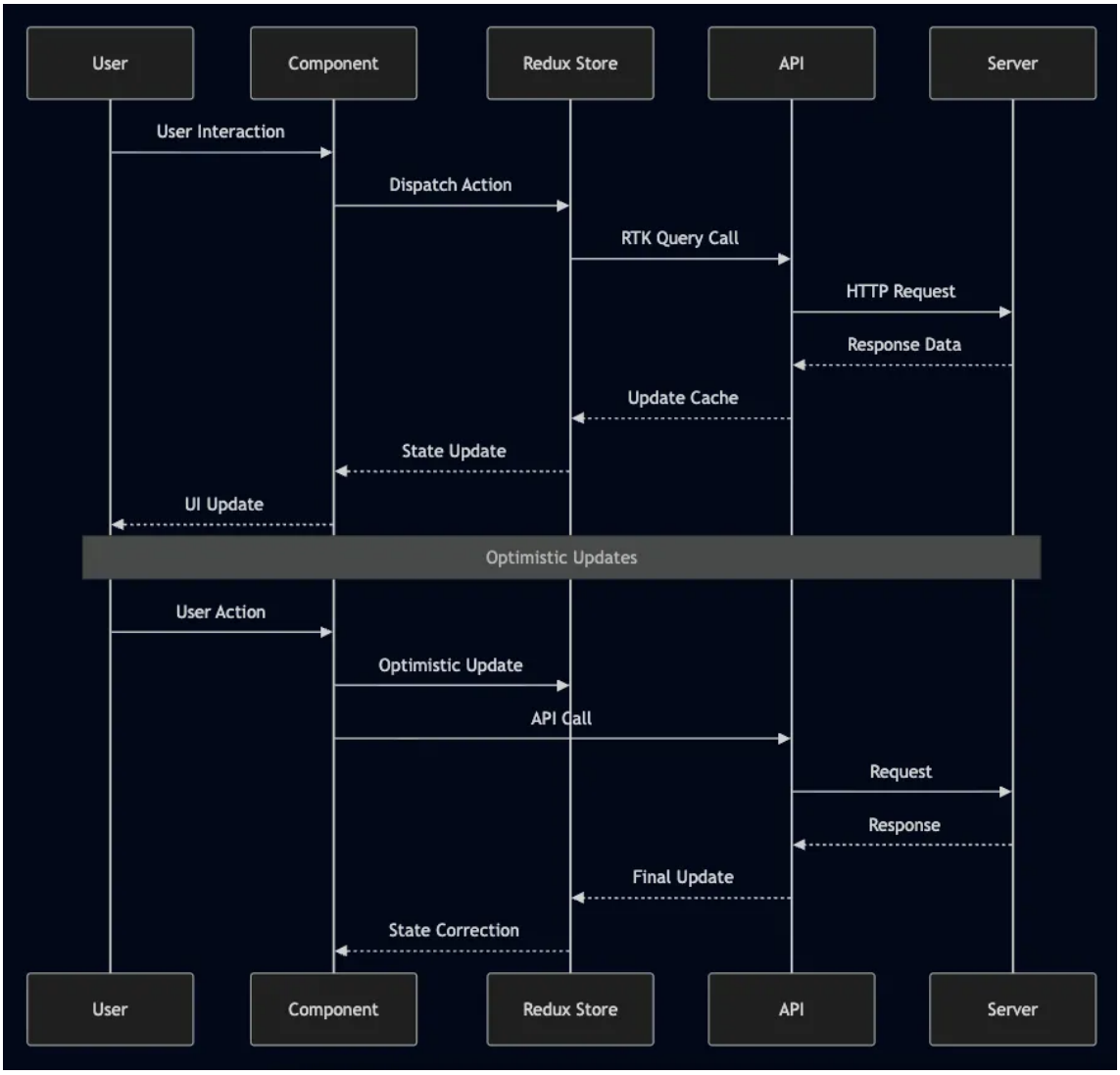
Container-Component Pattern

```
// Container Component (Redux Connected)
const EventsContainer = ({ events, loading, error, fetchEvents, navigate }) => {
    useEffect(() => {
        fetchEvents();
    }, [fetchEvents]);

    const handleEventClick = (eventId) => {
        navigate(`/events/${eventId}`);
    };
```

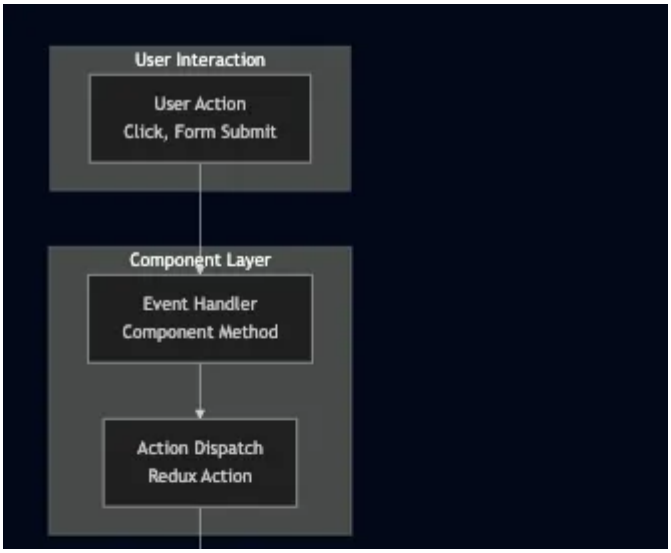
```
    return (
        <EventsList
            events={events}
            loading={loading}
            error={error}
            onEventClick={handleEventClick}
        />
    );
};

// Presentation Component (Pure)
const EventsList = ({ events, loading, error, onEventClick }) => {
    if (loading) return <Loader />;
    if (error) return <ErrorMessage error={error} />;

    return (
        <div className="events-list">
            {events.map((event) => (
                <EventCard
                    key={event.id}
                    event={event}
                    onClick={() => onEventClick(event.id)}
                />
            ))}
        </div>
    );
};
```

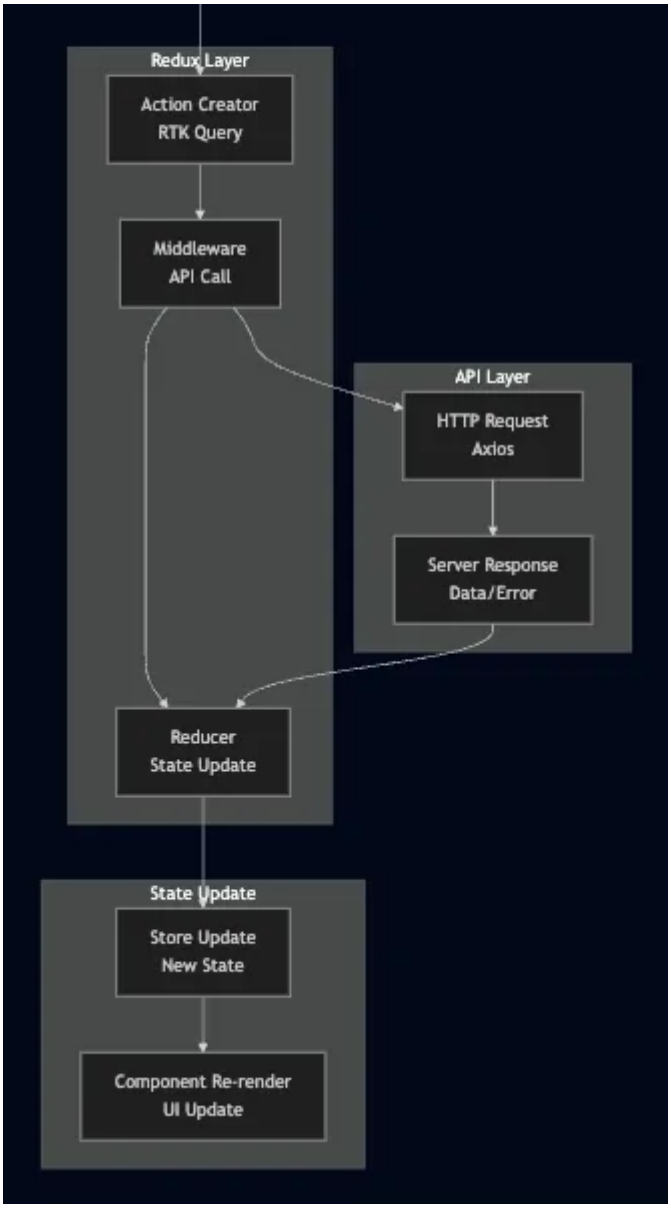
Data Flow

Application Data Flow



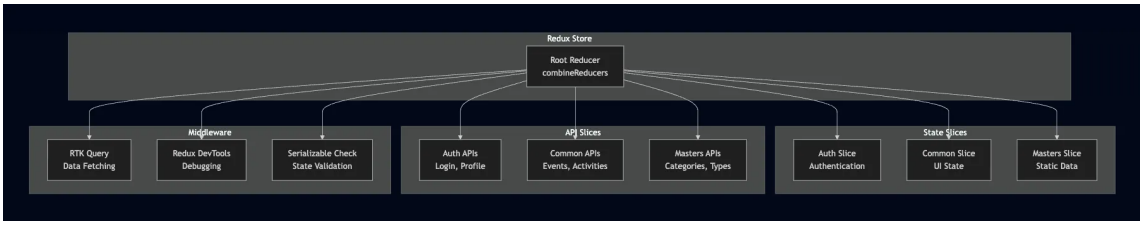
State Management Flow





🔄 State Management

Redux Architecture

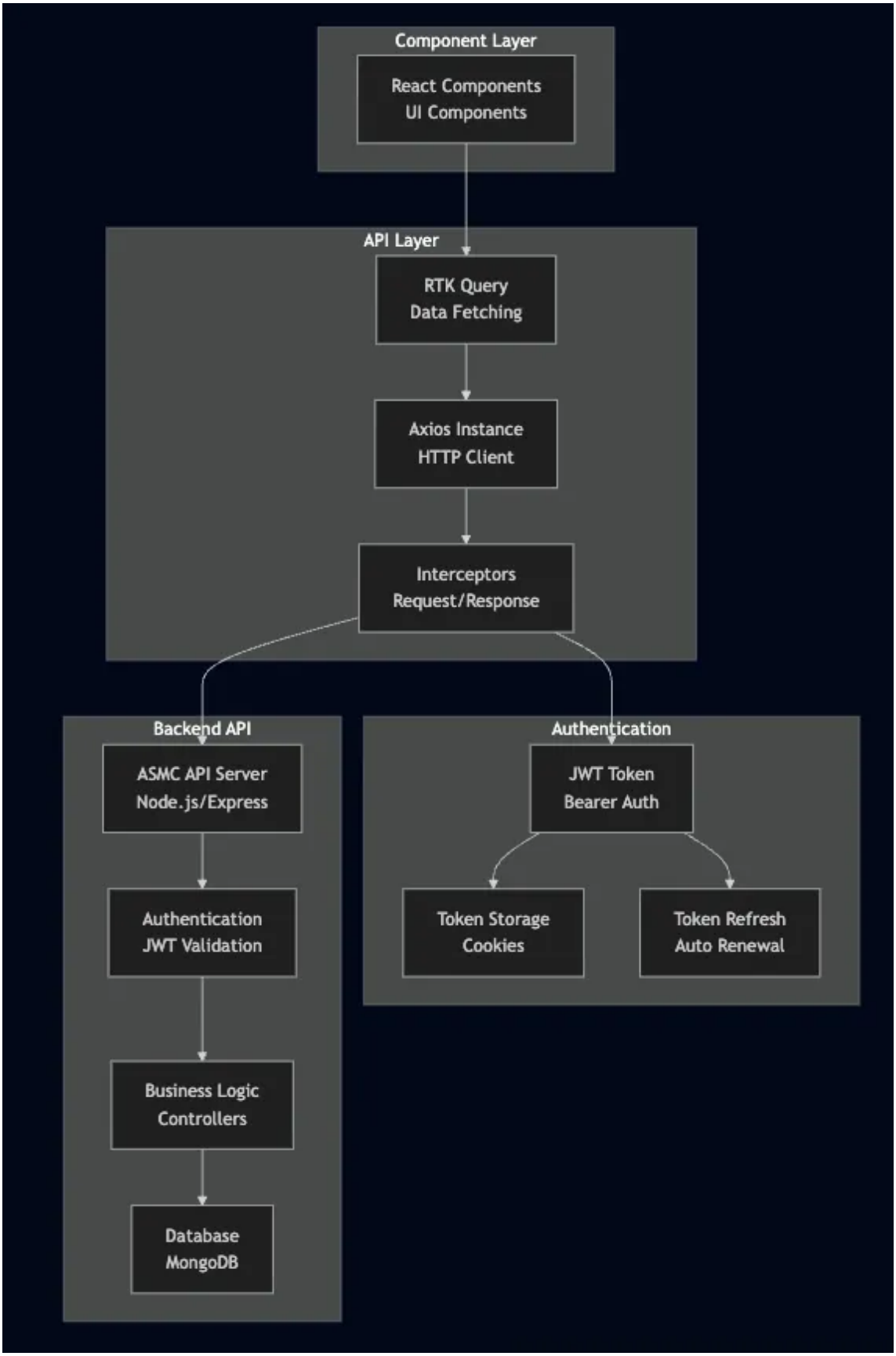


State Structure

```
// Root State Structure
{
    auth: {
        isAuthenticated: boolean,
        user: User | null,
        token: string | null,
        loading: boolean,
        error: string | null
    },
    common: {
        theme: 'light' | 'dark',
        language: string,
        notifications: Notification[],
        loading: boolean,
        modals: {
            login: boolean,
            register: boolean,
            payment: boolean
        }
    },
    masters: {
        categories: Category[],
        types: Type[],
        facilities: Facility[],
        loading: boolean
    },
    // RTK Query API States
    authApis: {
        queries: {},
        mutations: {},
        subscriptions: {}
    },
    commonApis: {
        queries: {},
        mutations: {},
        subscriptions: {}
    },
    mastersApis: {
        queries: {},
        mutations: {},
        subscriptions: {}
    }
}
```

API Integration

API Layer Architecture

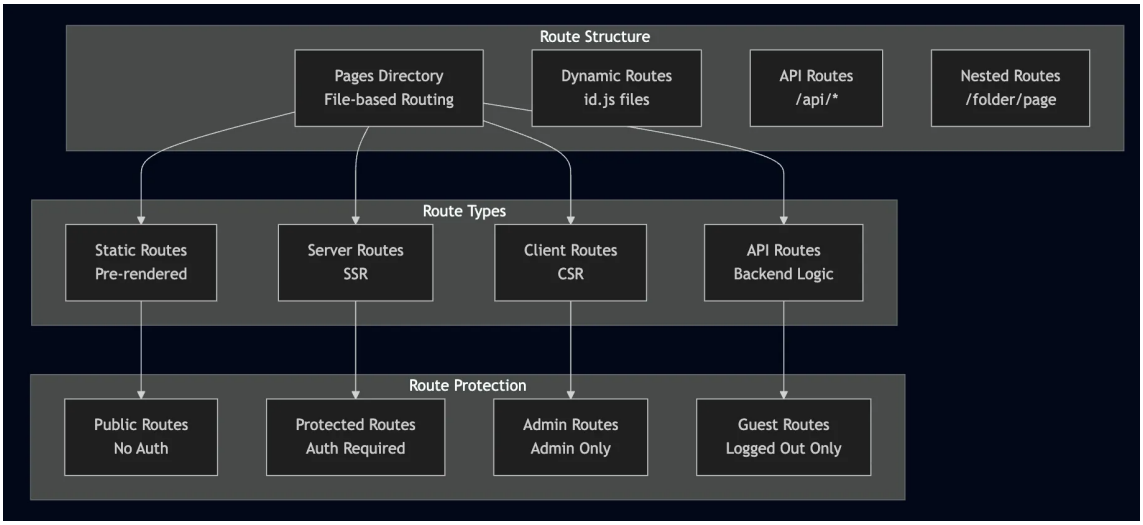

Component Layer
React Components
UI Components
API Layer
RTK Query
Data Fetching
Axios Instance
HTTP Client
Interceptors
Request/Response
Backend API
ASMC API Server
Node.js/Express
Authentication
JWT Validation
Business Logic
Controllers
Database
MongoDB
Authentication
JWT Token
Bearer Auth
Token Storage
Cookies
Token Refresh
Auto Renewal

API Configuration

```
// Axios Configuration
const axiosInstance = axios.create({
    baseURL: process.env.NEXT_PUBLIC_API_URL,
    timeout: 30000,
    headers: {
        'Content-Type': 'application/json',
    },
});

// Request Interceptor
axiosInstance.interceptors.request.use(
    (config) => {
        const token = getCookieData('token');
        if (token) {
            config.headers.Authorization = `Bearer ${token}`;
        }
        return config;
    },
    (error) => Promise.reject(error),
);

// Response Interceptor
axiosInstance.interceptors.response.use(
    (response) => response,
    (error) => {
        if (error.response?.status === 401) {
            removeCookieData('token');
            window.location.href = '/sign-in';
        }
        return Promise.reject(error);
    },
);
```

🔀 Routing Architecture

Next.js Pages Router

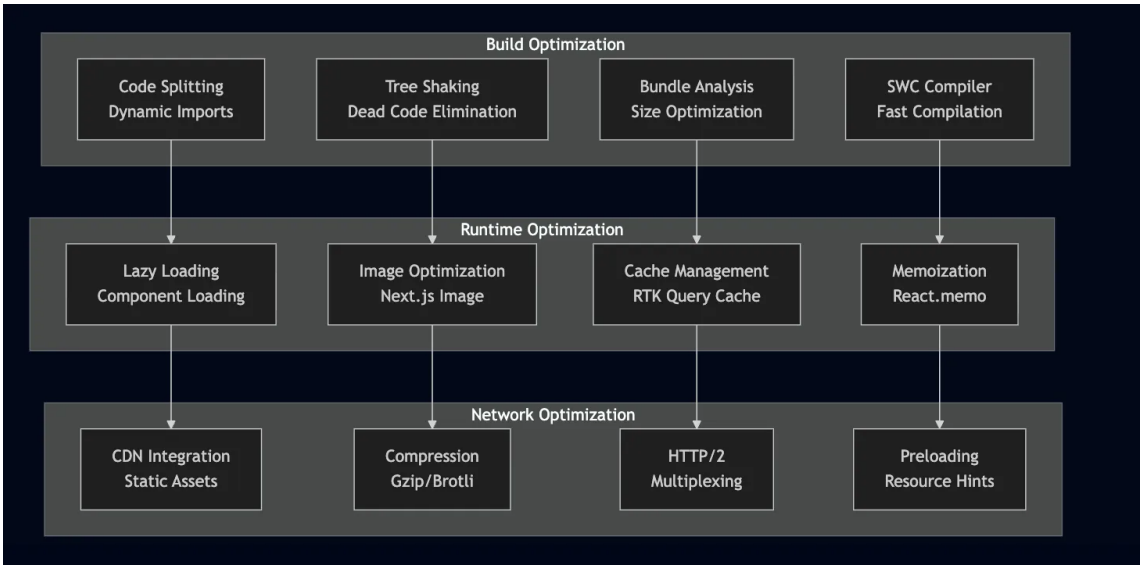


Route Configuration

```
// Dynamic Route Example
// pages/events/[event_id].js
import { useRouter } from 'next/router';
import { useGetEventQuery } from '@/redux/common/commonApis';

const EventDetails = () => {
    const router = useRouter();
    const { event_id } = router.query;

    const { data: event, isLoading, error } = useGetEventQuery(event_id);

    if (isLoading) return <div>Loading...</div>;
    if (error) return <div>Error loading event</div>;
    if (!event) return <div>Event not found</div>;

    return (
        <div>
            <h1>{event.title}</h1>
            <p>{event.description}</p>
        </div>
    );
};

export default EventDetails;
```

Performance Architecture

Performance Optimization



Performance Strategies

1. Code Splitting

```
// Dynamic imports for heavy components
import dynamic from 'next/dynamic';

const HeavyComponent = dynamic(() => import('../components/HeavyComponent'), {
    loading: () => <div>Loading...</div>,
    ssr: false,
});
```

2. Image Optimization

```
import Image from 'next/image';

<Image
    src="/images/hero.jpg"
    alt="Hero image"
    width={800}
    height={600}
    priority
    placeholder="blur"
    blurDataURL="..."
/>;
```

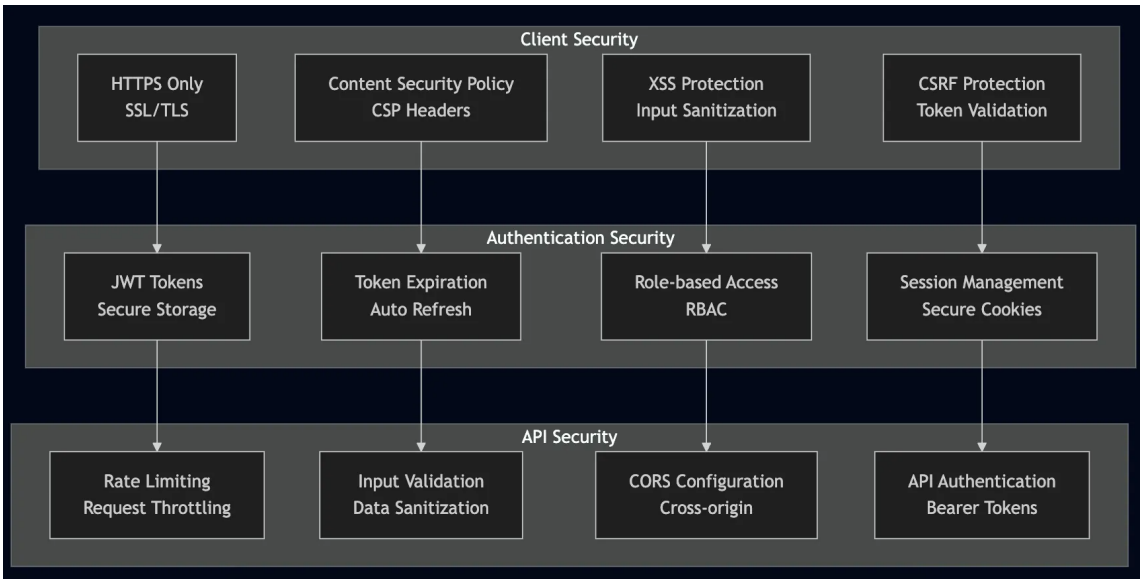
3. Cache Management

```
// RTK Query cache configuration
export const eventsApi = createApi({
    reducerPath: 'eventsApi',
    baseQuery: fetchBaseQuery({ baseUrl: '/api' }),
    tagTypes: ['Events'],
```

```
    endpoints: (builder) => ({
        getEvents: builder.query({
            query: () => 'events',
            providesTags: ['Events'],
            keepUnusedDataFor: 60, // 1 minute
        }),
    }),
});
```

🔒 Security Architecture

Security Layers



Security Implementation

1. Content Security Policy

```
// next.config.mjs
const nextConfig = {
    async headers() {
        return [
            {
                source: '/(.*)',
                headers: [
                    {
                        key: 'X-Frame-Options',
                        value: 'DENY',
                    },
                    {
                        key: 'X-Content-Type-Options',
                        value: 'nosniff',
                    },
                    {
```

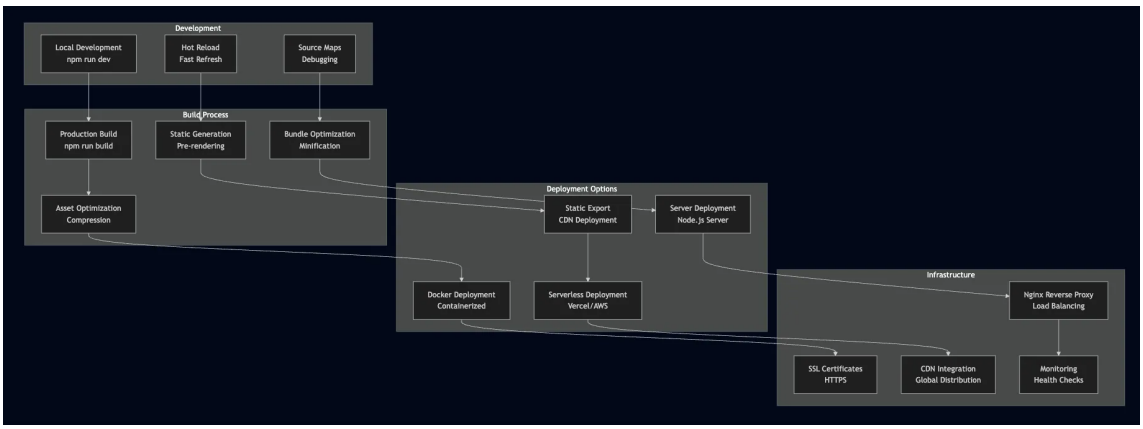
```
                        key: 'Referrer-Policy',
                        value: 'strict-origin-when-cross-origin',
                    },
                    {
                        key: 'Content-Security-Policy',
                        value: "default-src 'self'; script-src 'self' 'unsafe-eval'
'unsafe-inline'; style-src 'self' 'unsafe-inline'; img-src 'self' data: https:; font-
src 'self' data:;",
                    },
                ],
            },
        ];
    },
};
```

2. Authentication Security

```
// Token management
const setCookieData = (key, value, days = 7) => {
    const expires = new Date();
    expires.setTime(expires.getTime() + days * 24 * 60 * 60 * 1000);

    document.cookie =
`${key}=${value};expires=${expires.toUTCString()};path=/;secure;samesite=strict`;
};

const removeCookieData = (key) => {
    document.cookie = `${key}=;expires=Thu, 01 Jan 1970 00:00:00
UTC;path=/;secure;samesite=strict`;
};
```

Deployment Architecture

Deployment Strategy



Deployment Configuration

1. PM2 Configuration

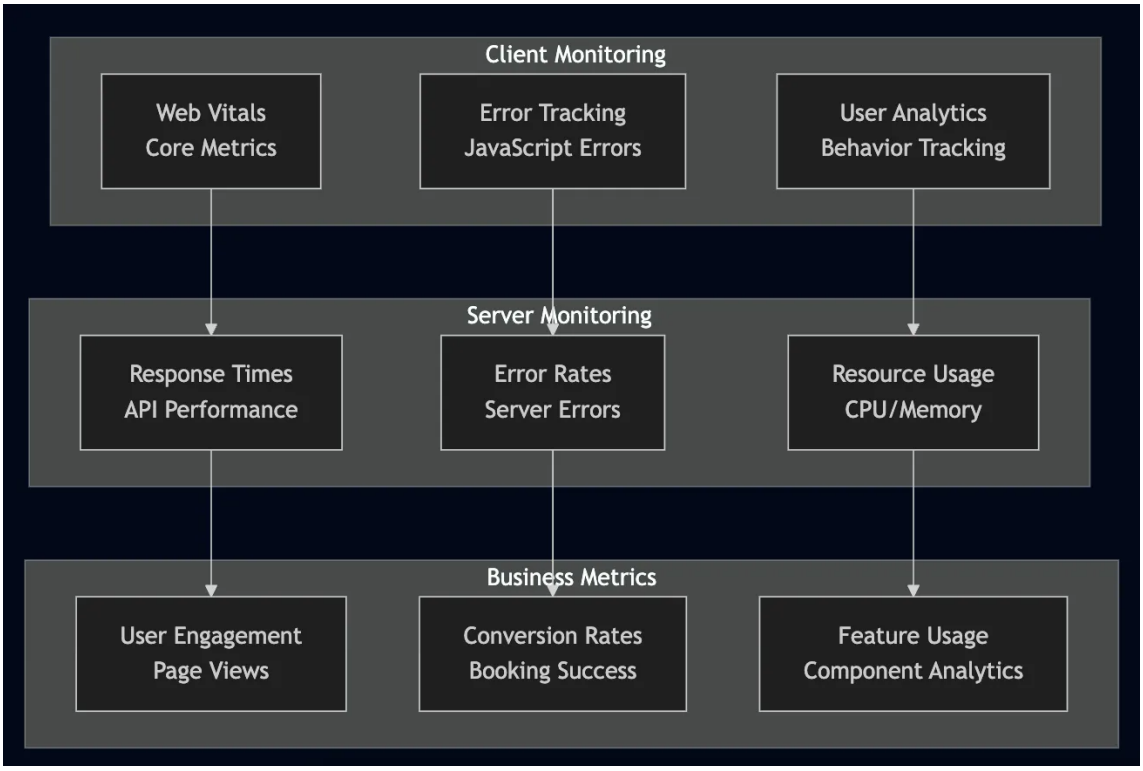
```
// ecosystem.config.js
module.exports = {
    apps: [
        {
            name: 'asmc-next',
            script: 'npm',
            args: 'start',
            instances: 'max',
            exec_mode: 'cluster',
            env: {
                NODE_ENV: 'production',
                PORT: 3000,
            },
            error_file: './logs/err.log',
            out_file: './logs/out.log',
            log_file: './logs/combined.log',
            time: true,
        },
    ],
};
```

2. Nginx Configuration

```
server {
    listen 80;
    server_name asmcdae.in;
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name asmcdae.in;

    ssl_certificate /etc/letsencrypt/live/asmcdae.in/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/asmcdae.in/privkey.pem;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_cache_bypass $http_upgrade;
    }
}
```

Monitoring & Analytics

Performance Monitoring



Monitoring Implementation

```
// Web Vitals tracking
import { getCLS, getFID, getFCP, getLCP, getTTFB } from 'web-vitals';

function sendToAnalytics(metric) {
    // Send to analytics service
    console.log(metric);
}

getCLS(sendToAnalytics);
getFID(sendToAnalytics);
getFCP(sendToAnalytics);
getLCP(sendToAnalytics);
getTTFB(sendToAnalytics);
```

Architecture Benefits

Scalability

- **Component-based architecture** enables easy feature additions
- **Redux state management** provides predictable state updates
- **RTK Query caching** reduces server load and improves performance

- **Next.js optimization** ensures fast page loads and SEO

Maintainability

- **Clear separation of concerns** between presentation and logic
- **Consistent coding patterns** across the application
- **Comprehensive documentation** for easy onboarding
- **Type safety** with PropTypes and ESLint rules

Performance

- **Server-side rendering** for better SEO and initial load times
- **Code splitting** for smaller bundle sizes
- **Image optimization** for faster page loads
- **Caching strategies** for improved user experience

Security

- **JWT authentication** for secure user sessions
- **Input validation** to prevent XSS attacks
- **HTTPS enforcement** for secure data transmission
- **Content Security Policy** for additional protection

Last Updated: January 2025
Maintainer: ASMC Development Team