Industrial Engineering Journal
ISSN: 0970-2555
Volume : 54, Issue 7, July : 2025

# 3D MEDICAL EXAMINATION VISUALIZER:

**Ayush Mohanty** (2101298204), 4th Year, Department of CSE (AI), Gandhi Institute for Technology, BPUT, INDIA amohanty2021@gift.edu.in
**Shubham Puhan** (2101298243), 4th Year, Department of CSE (AI), Gandhi Institute for Technology, BPUT, INDIA spuhan2021@gift.edu.in

**Abstract :-**
Manasvikaar is a modern, high-performance web application designed to deliver interactive and immersive user experiences through a robust frontend technology stack. Built using React 18, Vite, Three.js, and Material-UI (MUI), the project integrates 3D rendering, responsive UI components, and utility-first styling to support a wide range of creative and functional web applications. Key libraries such as React Three Fiber, Theatre.js, and React Router enable advanced animation control, seamless client-side routing, and gesture-based interactions. The development workflow is optimized through Vite's hot module replacement, ESLint integration, and support for utility tools like Tailwind CSS and Styled Components. Manasvikaar's modular architecture and diverse set of dependencies make it suitable for use cases like 3D portfolios, interactive dashboards, simulations, and creative coding projects. With future enhancements including TypeScript integration, backend connectivity, and cloud deployment, the project is well-positioned for scalability and real-world adoption.

## PROJECT OVERVIEW :-

### A Modern React Application with Vite, Three.js, and MUI

### 1. Introduction

The project **Manasvikaar** is a modern web application built using **React**, **Vite**, **Three.js**, and **MaterialUI (MUI)**. It leverages cutting-edge frontend technologies to deliver a high-performance, interactive, and visually appealing user experience. This report provides a detailed breakdown of the project's structure, dependencies, configurations, and potential use cases.

### 2. Overview :-

#### 2.1 Key Features

- **React 18** for building dynamic UIs with hooks and functional components.
- **Vite** as the build tool for fast development and optimized production builds.
- **Three.js & React Three Fiber** for 3D rendering and animations.
- **Material-UI (MUI)** for a polished and responsive UI design.
- **Theatre.js** for advanced animation control.
- **React Router DOM** for client-side routing.
- **Styled Components & Emotion** for CSS-in-JS styling.
- **Tailwind CSS** (via PostCSS) for utility-first styling.
- **React Hot Toast** for notifications.
- **React Use Gesture** for gesture-based interactions.

#### 2.2 Project Structure

The project follows a standard Vite + React structure:

```
Download
├── .gitignore
├── index.html
├── package.json
└── (expected directories: src/, public/)
```

## 3. Detailed File Analysis

### 3.1 .gitignore

This file specifies which files and directories should be excluded from version control:

- **Logs & Debug Files**:
  - *.log, npm-debug.log*, yarn-error.log*, etc.
  - Prevents cluttering the repository with temporary debug logs.
- **Build Artifacts**:
  - dist/, dist-ssr/ – Vite output directories.
- **Environment & Local Files**:
  - *.local – Local configuration files.
- **Editor & IDE Files**:
  - .vscode/, .idea/, .DS_Store, *.suo – Avoids IDE-specific files.

### 3.2 index.html

The entry point of the application:

- **Basic HTML5 Structure**:
  - Uses <div id="root"> for React mounting.
  - Includes a vite.svg favicon.
- **Vite Integration**:
  - <script type="module" src="/src/main.jsx"> – Loads the React app as an ES module.
  - Optimized for **Vite's HMR (Hot Module Replacement)**.

### 3.3 package.json

The heart of the project's configuration, defining dependencies, scripts, and metadata.

#### 3.3.1 Scripts

| Script | Description |
|---|---|
| dev | Runs Vite in development mode with HMR. |
| build | Generates an optimized production build in dist/. |
| lint | Runs ESLint for code quality checks (JS/JSX). |
| preview | Serves the production build locally for testing. |

#### 3.3.2 Dependencies

| Dependency | Purpose |
|---|---|
| react, react-dom | Core React libraries (v18). |
| @mui/material | Material-UI for pre-built React components. |
| @emotion/react | CSS-in-JS solution used by MUI. |
| @react-three/fiber | React renderer for Three.js (WebGL). |
| @react-three/drei | Helpers for Three.js in React. |
| @theatre/core | Animation toolkit for interactive scenes. |
| react-router-dom | Client-side routing (v6). |
| styled-components | CSS-in-JS styling alternative. |
| react-hot-toast | Notification system. |

| | |
|---|---|
| react-use-gesture | Handles drag, pinch, and other gestures. |
| three | 3D graphics library (WebGL). |

### 3.3.3 Dev Dependencies

| Dependency | Purpose |
|---|---|
| @vitejs/plugin-react-swc | Vite plugin for React (using SWC for faster builds). |
| eslint | Linting for code quality. |
| tailwindcss, postcss | Utility-first CSS framework with PostCSS. |
| @types/react | TypeScript definitions for React. |

## 4. Technology Stack Breakdown

### 4.1 Frontend Framework: React 18

- Uses **Functional Components & Hooks**.
- Supports **Concurrent Rendering** for smoother UX.

### 4.2 Build Tool: Vite

- **Blazing Fast HMR** (Hot Module Replacement).
- **ESM (ES Modules) by Default** – Optimized for modern browsers.
- **Pre-configured for React** via @vitejs/plugin-react-swc. **4.3 3D Rendering: Three.js & React Three Fiber**
- **React Three Fiber (R3F)** simplifies Three.js integration.
- **Drei** provides pre-built 3D components (e.g., cameras, lights).
- **Theatre.js** allows for timeline-based animations.

### 4.4 UI & Styling

- **Material-UI (MUI v5)** – Google's design system components.
- **Styled Components & Emotion** – CSS-in-JS for dynamic styling.
- **Tailwind CSS** – Utility-first CSS framework.

### 4.5 Additional Libraries

- **React Router v6** – Declarative routing.
- **React Hot Toast** – Toast notifications.
- **React Use Gesture** – Drag, pinch, and scroll interactions.

## 5. Potential Use Cases

Given the tech stack, this project could be:

1. **A 3D Portfolio Website** – Using Three.js for immersive visuals.
2. **An Interactive Dashboard** – MUI for UI, Theatre.js for animations.
3. **A Game or Simulation** – React Three Fiber for WebGL rendering.
4. **A Creative Coding Project** – Combining Theatre.js and Three.js.

## 6. Development Workflow

1. **Run** npm run dev → Starts Vite dev server.
2. **Code in** src/ – React components, 3D scenes, routes.
3. **Lint with** npm run lint – Ensures code quality.
4. **Build with** npm run build → Generates optimized assets in dist/.
5. **Preview with** npm run preview → Tests production build locally.

7. **Optimization & Best Practices** • **Vite's Built-in Optimizations**: o Code splitting, lazy loading, ESM support.
   • **Three.js Performance**:
      o Use @react-three/drei optimizations (e.g., Instances).
   • **MUI Theming**: o Customize themes with ThemeProvider.
   • **ESLint Enforcement**:
      o Prevents common React anti-patterns.

8. **Future Enhancements**
   • **Add TypeScript** → Better type safety.
   • **Integrate a Backend** → Firebase, Express, or Next.js API routes.
   • **Deploy to Vercel/Netlify** → CI/CD for automatic deployments.

9. **Conclusion**

**Manasvikaar** is a robust, modern web application leveraging **React, Vite, Three.js, and MUI** for highperformance interactivity. Its well-structured dependencies and build setup make it scalable for both simple and complex use cases, from 3D visualizations to interactive dashboards.

**Next Steps**:

- Explore adding **TypeScript**.
- Implement a **backend API**.
- Deploy to a cloud platform.