GLOBAL NEXT CONSULTING INDIA PRIVATE LIMITED

[March(2026) - May(2026)]

INTERNSHIP PROJECT REPORT

Full-Stack Web Development

React | Spring Boot | MongoDB

Submitted By:

Vivek Kumar Yadav

Under The Guidance Of –

Mr. Vansh (Mentor & Technical Head)

Submitted To –

GNCIPL

DECLARATION

I, Vivek Kumar Yadav, hereby declare that this internship report titled "Full-Stack Web Development Internship Report" is an authentic record of my own work carried out under the guidance of [Supervisor Name] at [Organization Name] during the period [Start Date] to [End Date].

The work presented in this report has not been submitted elsewhere for any academic or professional qualification. All sources of information have been duly acknowledged.

Signature: Vivek Kumar Yadav

Name: Vivek Kumar Yadav

ACKNOWLEDGMENT

I would like to express my sincere gratitude to [Organization Name] for providing me the opportunity to undertake this internship and gain practical exposure to full-stack web development.

I extend my heartfelt thanks to [Supervisor/Mentor Name] for their continuous guidance, technical mentorship, and constructive feedback throughout the internship period. Their insights were invaluable in shaping the projects documented in this report.

I am also grateful to [College/University Name] for encouraging industry exposure as part of the [Degree Program Name] curriculum, and to my peers for their constant support and collaboration.

Finally, I thank everyone at [Organization Name] who contributed, directly or indirectly, to the successful completion of this internship.

Vivek Kumar Yadav

[15/05/2026]

ABSTRACT

This report documents the comprehensive full-stack web development work completed during a structured internship at [Organization Name]. The program followed a progressive weekly curriculum — beginning with foundational frontend technologies and advancing to complex backend systems — culminating in the development of Matty, a production-ready graphic design platform.

Weekly Project Summary

Over five weeks, the following projects were developed:

- Week 1 — A responsive portfolio website built with semantic HTML5 and modular CSS architecture.
- Week 2 — A React-based Todo application with state management via hooks and localStorage persistence.
- Week 3 — An advanced React lab exploring Vite build tooling, Tailwind CSS, and modern React 19 patterns.
- Week 4 — A production-ready Blog API using Spring Boot, MongoDB, and JWT authentication.
- Week 5 — A Task Manager backend API leveraging Spring Data JPA with an H2 in-memory database.

Capstone Project — Matty

The capstone project, Matty, is a full-stack graphic design platform combining a React-based canvas editor (powered by Fabric.js) with a robust Spring Boot backend, MongoDB for metadata storage, and Cloudinary for cloud asset management. The platform features JWT-based authentication, a glassmorphic dark-themed UI, template library, project dashboard, and role-based admin controls.

Key Achievements

- Delivered 5 functional mini-projects demonstrating progressive skill development.
- Built and deployed a production-ready full-stack application (Matty).
- Gained proficiency in React 19, Spring Boot 3, MongoDB, and Cloudinary integration.
- Implemented secure authentication using JWT and Spring Security.
- Successfully deployed applications using Vercel and cloud platforms.

CHAPTER 1: INTRODUCTION

1.1 Internship Overview

This internship was undertaken at [Organization Name] from [Start Date] to [End Date] as part of the [Degree Program Name] curriculum. The primary objective was to gain hands-on, industry-level experience in full-stack web development using modern technologies including React, Spring Boot, MongoDB, and cloud services.

The internship followed a structured weekly approach — beginning with foundational frontend development and progressively advancing through complex backend systems to a fully integrated capstone project.

1.2 Organization Profile

[Organization Name] is a technology-driven organization specializing in web and mobile application development. The organization places strong emphasis on innovation, code quality, and practical skill development, providing interns with real-world project experience.

1.3 Internship Objectives

#	Objective	Status
O1	Develop proficiency in modern frontend frameworks (React)	☑ Achieved
O2	Master backend development with Spring Boot	☑ Achieved
O3	Design and implement RESTful APIs	☑ Achieved
O4	Implement secure authentication mechanisms (JWT)	☑ Achieved
O5	Integrate databases (MongoDB, H2) with applications	☑ Achieved
O6	Build and deploy a complete full-stack application	☑ Achieved
O7	Learn cloud storage integration (Cloudinary)	☑ Achieved

1.4 Skill Progression Overview

Phase	Technology Focus	Output
Week 1	HTML5, CSS3, JavaScript	Portfolio Website
Week 2	React 19, Vite, Tailwind CSS	Todo Application
Week 3	Advanced React, Service Layers	React Task Manager Lab
Week 4	Spring Boot, MongoDB, JWT	Blog REST API
Week 5	Spring Boot, H2, JPA	Task Manager API
Capstone	React + Spring Boot + MongoDB + Cloudinary	Matty — Design Platform

CHAPTER 2: WEEKLY PROJECT DOCUMENTATION

2.1 Week 1 — Portfolio Website

Attribute	Details
Project Name	Portfolio Website
GitHub	github.com/vivekYadav129/portfolio
Deployment	portfolio-five-sandy-64.vercel.app
Technologies	HTML5, CSS3, JavaScript (ES6+), Font Awesome, Google Fonts

Purpose

To establish a professional online presence showcasing skills, projects, and contact information using modern, responsive web design principles.

Key Features & Technical Implementation

Feature	Description
Modular CSS Architecture	CSS split into logical files (base, header, hero, about, skills, projects, contact, footer) imported via main.css
CSS Custom Properties	Centralized theming using :root variables for colors, spacing, and animation parameters
Responsive Design	Three breakpoints: Mobile (<768px), Tablet (768–1024px), Desktop (>1024px)
Smooth Animations	Keyframe animations for fade-in effects and hover transitions
Semantic HTML5	Proper use of <header>, <main>, <section>, <article>, <footer> tags
DOM Manipulation	Vanilla JavaScript for mobile menu toggle and form handling

Learning Outcomes

Skill Area	Proficiency Gained
Semantic HTML & SEO	Understanding of accessibility and SEO best practices
CSS Grid & Flexbox	Mastery of modern layout techniques
Responsive Design	Mobile-first approach using media queries
CSS Variables	Dynamic theming and maintainable stylesheet organization
DOM Manipulation	Vanilla JavaScript for interactive UI elements
Version Control	Git workflow with structured, meaningful commits

2.2 Week 2 — Todo Application

Attribute	Details
Project Name	Todo App
GitHub	github.com/vivekYadav129/todo-app
Deployment	todo-app-one-pi-60.vercel.app
Technologies	React 19, Vite, Tailwind CSS, ESLint

Purpose

To build an interactive task management application demonstrating React fundamentals, state management with hooks, and browser-side data persistence.

Core Features

Feature	Technical Implementation
Add Tasks	Form handling with useState and controlled components
Complete / Incomplete Toggle	Immutability-safe state mutation pattern
Delete Tasks	Array filter method for item removal
LocalStorage Persistence	useEffect hooks to save/load tasks on mount and state change
Responsive UI	Tailwind CSS utility classes with mobile-first design
Fast Development	Vite HMR for near-instant hot module replacement

Learning Outcomes

Skill Area	Proficiency Gained
React Hooks	Practical use of useState and useEffect
Component Architecture	Breaking UI into reusable, composable components
Controlled Components	Form input management in React
LocalStorage API	Client-side data persistence
Tailwind CSS	Utility-first styling workflow
Vite Build Tool	Fast dev server setup and optimized production builds

2.3 Week 3 — React Project Lab (Vite + Task Manager)

Attribute	Details
Project Name	React Project Lab — Vite + Task Manager
GitHub	github.com/vivekYadav129/react-project
Technologies	React 19, Tailwind CSS, Vite, PostCSS, ESLint

Purpose

To explore advanced React patterns, Vite build tooling configuration, and service-layer architecture through a comprehensive task manager laboratory application.

Key Technical Highlights

Feature	Description
Vite Build Pipeline	Ultra-fast HMR, optimized production builds, esbuild pre-bundling
Service Layer Pattern	Clean separation of business logic from UI components
Custom React Hooks	Extracted reusable logic into dedicated hook files
ESLint Configuration	Code quality enforcement with React-specific rules
Component Composition	Reusable, props-driven component hierarchy
React 19 Features	Concurrent rendering capabilities and latest API patterns

Vite vs. Create React App — Performance Comparison

Aspect	Vite	Create React App
Dev Server Start	~300ms	~5–10 seconds
HMR Speed	Near-instant	Slower on large apps
Build Time	2–3x faster	Slower
Configuration	Simple and extensible	Black-box, eject required
Bundle Size	Smaller optimized chunks	Larger default bundles

2.4 Week 4 — Blog API (Spring Boot + MongoDB + JWT)

Attribute	Details
Project Name	Blog API
GitHub	github.com/vivekYadav129/product-api
Technologies	Spring Boot 3, Java 17, MongoDB, Spring Security, JJWT, Maven, Lombok

Purpose

To build a production-ready RESTful API for a blogging platform with secure JWT-based authentication, MongoDB persistence, and a clean layered architecture.

API Endpoints

Method	Endpoint	Description	Auth Required
POST	/auth/register	Register new user	None
POST	/auth/login	Login and receive JWT token	None
GET	/posts	Retrieve all posts	Optional

Method	Endpoint	Description	Auth Required
GET	/posts/{id}	Retrieve a specific post	Optional
POST	/posts	Create a new post	JWT Required
PUT	/posts/{id}	Update an existing post	JWT Required
DELETE	/posts/{id}	Delete a post	JWT Required

Key Security Components

Component	Class	Responsibility
JWT Filter	JwtAuthenticationFilter	Extract token from header, validate, set security context
JWT Utility	JwtUtils	Token generation, parsing, and signature/expiry validation
User Details	CustomUserDetailsService	Load user from MongoDB during authentication
Security Config	SecurityConfig	Define public/private endpoints, CORS, and password encoder

2.5 Week 5 — Task Manager Backend (Spring Boot + H2 + JPA)

Attribute	Details
Project Name	Task Manager Backend
GitHub	github.com/vivekYadav129/task-manager
Technologies	Spring Boot 3.2.5, Java 17, H2 Database, Spring Data JPA, Maven, Lombok

Purpose

To build a lightweight, fast-setup task management backend demonstrating ORM concepts and JPA repositories using an H2 in-memory database — ideal for rapid development and testing environments.

API Endpoints

Method	Endpoint	Description
GET	/tasks	Retrieve all tasks
GET	/tasks/{id}	Retrieve a task by ID
POST	/tasks	Create a new task
PUT	/tasks/{id}	Update an existing task
DELETE	/tasks/{id}	Delete a task by ID

Week 4 vs. Week 5 — Comparative Analysis

Aspect	Week 4: Blog API	Week 5: Task Manager
Database	MongoDB (NoSQL)	H2 In-Memory (SQL / JPA)
Authentication	JWT + Spring Security	None — Open API
ORM Layer	None (MongoRepository)	Spring Data JPA / Hibernate
Use Case	Production-ready service	Development and testing
Complexity	Higher (security layer)	Lower (ORM-focused)

CHAPTER 3: CAPSTONE PROJECT — MATTY

3.1 Project Overview

Matty is a state-of-the-art, full-stack graphic design platform developed as the capstone project of this internship. It provides a professional-grade, browser-based design studio combining a powerful Fabric.js canvas editor with cloud-based asset management, JWT-secured user authentication, project organization, and an extensible template library.

Core Purpose

To democratize professional graphic design by offering a powerful yet accessible alternative to expensive desktop software (e.g., Adobe Photoshop) and feature-limited free tools (e.g., Canva free tier). Matty enables creators, marketers, and entrepreneurs to produce high-quality visual content directly in their browser — with zero installation and full cloud support.

Target Users

User Type	Primary Use Cases
Graphic Designers	Creating social media graphics, posters, and promotional flyers
Marketers	Designing ad creatives, banners, and email headers
Entrepreneurs	Building brand assets and product mockups
Students	Learning design principles and creating project presentations

3.2 Problem Statement

Limitations of Existing Solutions

Solution	Key Limitation
Adobe Photoshop	Not cloud-native; requires installation and powerful hardware; expensive subscription
Canva (Free Tier)	Watermarks, restricted assets, and advanced features locked behind paywall
Figma	Primarily UI/UX focused; not suited for general graphic design workflows
Raw Fabric.js Demos	No backend, no authentication, no project management or asset organization

How Matty Addresses These Limitations

Problem	Matty's Solution
High cost & steep learning curve	Zero-cost, browser-based access with an intuitive modern UI
No cloud-native storage	Full cloud integration via Cloudinary for persistent asset management

Problem	Matty's Solution
Poor user experience	Glassmorphic dark theme with clean, responsive interface design
Performance on complex designs	Vite + Fabric.js optimization for smooth canvas operations
No backend / user management	Complete Spring Boot backend with JWT-secured user accounts
Scalability concerns	Stateless JWT auth and decoupled architecture ready for horizontal scaling

3.3 System Architecture

Three-Tier Architecture

Tier	Component	Technology
Presentation Tier	React Single-Page Application (SPA)	React 19 + Vite + Tailwind CSS
Presentation Tier	Canvas Design Editor	Fabric.js
Presentation Tier	State Management	Redux Toolkit
Application Tier	REST API Server	Spring Boot 3.2.5 (Java 17)
Application Tier	Authentication & Authorization	Spring Security + JWT (JJWT)
Data Tier	Metadata & User Data	MongoDB Atlas
Storage Tier	Image & Asset Storage	Cloudinary CDN

End-to-End Upload Workflow

The following table describes the complete request lifecycle when a user uploads an image to their design:

Step	Action	Component
1	User clicks 'Upload Image'	React Editor UI
2	React creates FormData and sends POST to /api/upload/image with Bearer JWT	Axios + API Service
3	JWT Authentication Filter validates the token	JwtAuthenticationFilter
4	Controller receives request and delegates to CloudinaryService	UploadController
5	Cloudinary SDK uploads the file and returns a public CDN URL	CloudinaryService
6	Backend saves the asset URL to the user's project in MongoDB	DesignRepository

Step	Action	Component
7	Backend returns the asset URL in the HTTP response	ResponseEntity
8	React creates a Fabric.js image object and places it on the canvas	Fabric.js API
9	User edits the object (resize, reposition, filters) — all local operations	Fabric.js Canvas
10	User clicks 'Save' — frontend serialises canvas JSON and sends PUT request	PUT /api/designs/{id}

3.4 Technology Stack

Frontend

Technology	Version	Role
React	19	Component-based UI; efficient rendering via virtual DOM
Vite	Latest	Next-generation build tool; 10x faster HMR than Create React App
Tailwind CSS	Latest	Utility-first framework for rapid, consistent UI development
Redux Toolkit	Latest	Predictable state management for auth, projects, and UI state
Fabric.js	Latest	Powerful canvas engine with object model, serialization, and event handling
Lucide React	Latest	Modern, tree-shakable icon set

Backend

Technology	Version	Role
Spring Boot	3.2.5	Production-ready Java framework with auto-configuration and embedded Tomcat
Java	17 (LTS)	Strong typing, performance, and an extensive enterprise ecosystem
Spring Security	6.x	Industry-standard authentication and authorization framework
JJWT	0.11.5	JWT token creation, parsing, and signature validation
Maven	Latest	Dependency management and build automation
Lombok	Latest	Boilerplate reduction via annotations (@Data, @Builder, etc.)

Database & Cloud Services

Technology	Role	Rationale
MongoDB	Metadata storage	NoSQL schema flexibility ideal for JSON canvas state; horizontal scaling

Technology	Role	Rationale
Cloudinary	Asset storage & delivery	CDN-based delivery, automatic image optimization, and transformation APIs

3.5 Database Design

Collections Overview

Matty uses three primary MongoDB collections:

- User Collection — Stores account credentials, roles, profile metadata, and login timestamps.
- Design Collection — Stores the complete Fabric.js canvas JSON state, thumbnail URLs, and asset references for each user's design project.
- Template Collection — Stores reusable community/admin-created templates with category metadata, thumbnail, and usage analytics.

Indexing Strategy

Collection	Index Field	Index Type	Purpose
User	username	Unique	Fast lookup during login
User	email	Unique	Prevent duplicate account registration
Design	userId	Single	Efficiently list all designs belonging to a user
Design	createdAt	Descending	Sort designs by most recently created
Design	tags	Multikey	Filter designs by tag categories
Template	category	Single	Filter template library by design category

3.6 API Documentation

Authentication Endpoints

Method	Endpoint	Description	Auth
POST	/api/auth/register	Register a new user account	None
POST	/api/auth/login	Authenticate user and receive JWT token	None

Design Endpoints

Method	Endpoint	Description	Auth
GET	/api/designs	Get all designs for the authenticated user	Bearer JWT
GET	/api/designs/{id}	Get a specific design by ID	Bearer JWT
POST	/api/designs	Create a new design	Bearer JWT

Method	Endpoint	Description	Auth
PUT	/api/designs/{id}	Update an existing design (title + canvas state)	Bearer JWT
DELETE	/api/designs/{id}	Delete a design (owner or admin only)	Bearer JWT

Upload & Admin Endpoints

Method	Endpoint	Description	Auth
POST	/api/upload/image	Upload image to Cloudinary; returns CDN URL	Bearer JWT
GET	/api/admin/stats	Platform statistics (users, designs, storage)	ADMIN Role
GET	/api/admin/users	Paginated user list with sort options	ADMIN Role

3.7 Security Implementation

Authentication Flow

Matty uses a stateless JWT-based authentication mechanism:

1. User submits credentials via POST /api/auth/login.
2. CustomUserDetailsService loads the user record from MongoDB.
3. BCryptPasswordEncoder (strength=10) validates the submitted password against the stored hash.
4. JwtUtils generates a signed HS512 JWT token containing the user ID and roles.
5. The token is returned to the client and stored in the browser (localStorage / Redux state).
6. All subsequent requests include the token as a Bearer header.
7. JwtAuthenticationFilter intercepts every request, validates the token, and populates the Spring Security context.
8. Secured endpoints are only accessible when a valid token is present and the user has the required role.

Security Headers Applied

Header	Value	Protection Purpose
X-Content-Type-Options	nosniff	Prevents MIME type sniffing attacks
X-Frame-Options	DENY	Protects against clickjacking
Cache-Control	no-cache, no-store, must-revalidate	Prevents caching of sensitive response data

3.8 Core Functionalities

1. Canvas Editor (Fabric.js)

The canvas editor is the central feature of Matty. It supports the following operations:

- Adding and editing text objects (IText) with full font and color control.

- Uploading images from local storage and placing them on the canvas with resize, reposition, and rotation support.
- Drawing geometric shapes (rectangles, circles, lines, paths).
- Layer management — objects can be brought forward, sent backward, grouped, or deleted.
- Full canvas state serialization to JSON (canvas.toJSON()) for persistence in MongoDB.
- Canvas state restoration from saved JSON on design load (canvas.loadFromJSON()).

2. Asset Management (Cloudinary)

All media assets are managed via Cloudinary:

- Images are uploaded to the backend via multipart/form-data POST requests authenticated with JWT.
- The Spring Boot CloudinaryService uses the Cloudinary Java SDK to upload files and apply transformations (e.g., size constraints).
- Returned public CDN URLs are stored in the design's assetList in MongoDB and rendered on the Fabric.js canvas.
- Cloudinary's global CDN ensures fast image delivery across geographies.

3. State Management (Redux Toolkit)

Redux Toolkit manages three primary state slices:

Slice	State Managed	Key Actions
authSlice	User identity, JWT token, authentication status	setCredentials, logout
designSlice	Current design data, canvas state, loading state	setDesign, updateCanvas, clearDesign
uiSlice	Modal visibility, sidebar state, active tool selection	openModal, toggleSidebar, setActiveTool

3.9 Advantages of Matty

Technical Advantages

Advantage	Description
Modern Technology Stack	React 19 + Vite + Spring Boot 3 + MongoDB + Cloudinary
High Performance	Vite for fast builds; Fabric.js optimized for smooth canvas rendering
Scalable Architecture	Decoupled frontend/backend, stateless JWT, horizontally scalable
Type Safety	Java backend eliminates runtime type errors
Production Ready	Full security, logging, error handling, and input validation implemented

User Benefits

Benefit	Description
Zero Installation	Browser-based — works on any device with an internet connection

Benefit	Description
Professional Quality	Advanced canvas features: layers, grouping, transformations, text editing
Cloud Storage	Designs accessible from anywhere; never lost
Beautiful UI	Glassmorphic dark theme reduces eye strain and delivers a premium experience

3.10 Limitations & Future Scope

Current Limitations

Limitation	Description	Priority to Address
No Real-time Collaboration	Currently single-user only; no shared editing	High
No AI Features	Roadmap items not yet implemented	High
Limited Export Formats	PDF and SVG export not yet available	Medium
No Undo/Redo UI	Fabric.js supports history but UI controls not yet exposed	Low
No Design Version History	Canvas is overwritten on every save; no rollback available	Medium

Planned Future Enhancements

Feature	Description	Priority
AI Design Suggestions	Layout generation via OpenAI/Gemini API; text-to-design prompts	High
Real-time Collaboration	Multi-user canvas editing using WebSocket and Operational Transforms	High
Export to PDF / SVG	Multiple format support for print and digital publishing	Medium
Design Version History	Track changes and allow canvas state restoration	Medium
Team Workspaces	Shared projects with role-based access control	Low
OAuth2 Integration	Google and GitHub single sign-on (SSO)	Low

CHAPTER 4: RESULTS & DISCUSSION

4.1 Project Completion Status

Component	Status	Completion (%)
Week 1: Portfolio Website	☑ Complete	100%
Week 2: Todo Application	☑ Complete	100%
Week 3: React Project Lab	☑ Complete	100%
Week 4: Blog API	☑ Complete	100%
Week 5: Task Manager API	☑ Complete	100%
Matty — Frontend	☑ Complete	95%
Matty — Backend	☑ Complete	95%
Matty — Database	☑ Complete	100%
Matty — Deployment	☑ Complete	90%

4.2 Key Achievements

- Successfully delivered 5 functional mini-projects demonstrating progressive skill development across frontend and backend domains.
- Built and deployed a production-ready full-stack application (Matty) from scratch — spanning UI design, REST API, database modelling, and cloud storage integration.
- Implemented industry-standard secure authentication using JWT and Spring Security with role-based access control.
- Integrated two distinct database technologies (MongoDB and H2/JPA) within different project contexts.
- Gained hands-on proficiency in React 19, Vite, Spring Boot 3, MongoDB, and Cloudinary.
- Deployed applications to Vercel and cloud platforms with production-grade configurations.

4.3 Challenges & Solutions

Challenge Encountered	Solution Applied
Understanding Spring Security & JWT integration	Systematic documentation review and step-by-step implementation with unit testing at each stage
Managing Fabric.js canvas state for persistence	Deep dive into Fabric.js API docs; implemented canvas.toJSON() and loadFromJSON() patterns
MongoDB schema design for nested JSON canvas data	Used MongoDB's flexible document model with embedded objects for canvas state
Cloudinary SDK integration with Spring Boot	Followed official SDK documentation and built a dedicated CloudinaryService abstraction layer

Challenge Encountered	Solution Applied
CORS errors between React frontend and Spring Boot backend	Configured CorsConfig.java with explicit allowed origins and methods in SecurityConfig

4.4 Skills Acquired

Technical Skills

Category	Skills
Frontend Development	React 19, Vite, Tailwind CSS, Redux Toolkit, Fabric.js, ES6+
Backend Development	Spring Boot 3, Spring Security, JWT, REST API Design, Java 17
Database & ORM	MongoDB, Spring Data JPA, H2, Hibernate
Cloud & Deployment	Cloudinary, Vercel, environment configuration
Tools & Practices	Git version control, Maven, Postman, ESLint, Lombok

Professional Skills

Skill	How Developed
Problem Solving	Debugging complex issues across the full stack — from CORS errors to JWT token expiry
Technical Documentation	Writing README files, API documentation, and this internship report
Time Management	Delivering working projects on a weekly schedule without missing milestones
Version Control Discipline	Structured Git commits with meaningful messages across all repositories

CHAPTER 5: CONCLUSION & FUTURE SCOPE

5.1 Conclusion

This internship provided comprehensive, hands-on exposure to the full spectrum of modern web development — from foundational frontend design to production-grade backend systems and cloud-integrated full-stack platforms. The structured weekly curriculum ensured progressive skill development, with each project building upon the concepts and tools introduced in the previous week.

The five weekly projects established a solid technical foundation across HTML/CSS, React, Vite, Spring Boot, REST APIs, JWT authentication, and both SQL and NoSQL databases. The capstone project, Matty, successfully integrated all of these skills into a coherent, production-ready application — demonstrating not just technical proficiency, but also sound architectural thinking, security awareness, and a focus on user experience.

The Matty platform exemplifies the following industry-relevant capabilities acquired during the internship:

- Complete full-stack architecture with clear separation of concerns across presentation, application, and data tiers.
- Secure, stateless authentication and authorization using JWT and Spring Security.
- Cloud-based asset management with Cloudinary for scalable media delivery.
- A professional, accessible UI built with React 19, Tailwind CSS, and Fabric.js.
- A scalable and maintainable codebase structured for future enhancement and team collaboration.

5.2 Future Scope

Immediate Enhancements

- Expose undo/redo controls in the Fabric.js editor using its built-in history management.
- Implement PDF and SVG export functionality for completed designs.
- Expand the template library with additional categories and professionally designed templates.
- Improve mobile responsiveness of the editor canvas for tablet and touch-screen users.

Advanced Features (Roadmap)

- AI Integration — layout suggestion and background removal via OpenAI/Gemini APIs; text-to-design prompt generation.
- Real-time Collaboration — multi-user concurrent canvas editing using WebSocket (Spring STOMP) and Operational Transforms.
- Design Version History — snapshot-based versioning to allow rollback to previous canvas states.
- Team Workspaces — shared project environments with role-based access control.

Enterprise Features

- OAuth2 integration for social login (Google, GitHub).
- API rate limiting and usage analytics for platform administrators.
- Payment integration for premium templates and advanced features.

CHAPTER 6: REFERENCES

6.1 Official Documentation

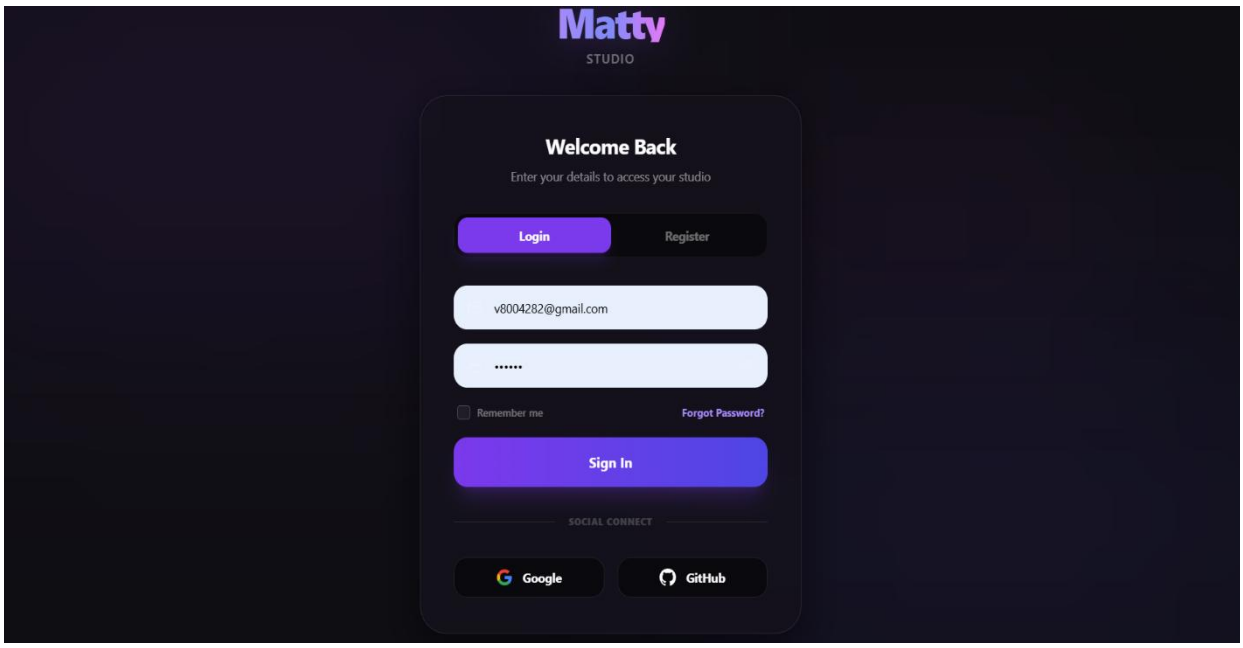
Technology	Reference
React 19	https://react.dev/
Vite	https://vitejs.dev/
Tailwind CSS	https://tailwindcss.com/
Redux Toolkit	https://redux-toolkit.js.org/
Fabric.js	http://fabricjs.com/
Spring Boot	https://spring.io/projects/spring-boot
Spring Security	https://spring.io/projects/spring-security
MongoDB	https://www.mongodb.com/docs/
Cloudinary	https://cloudinary.com/documentation
JJWT Library	https://github.com/jwtk/jjwt

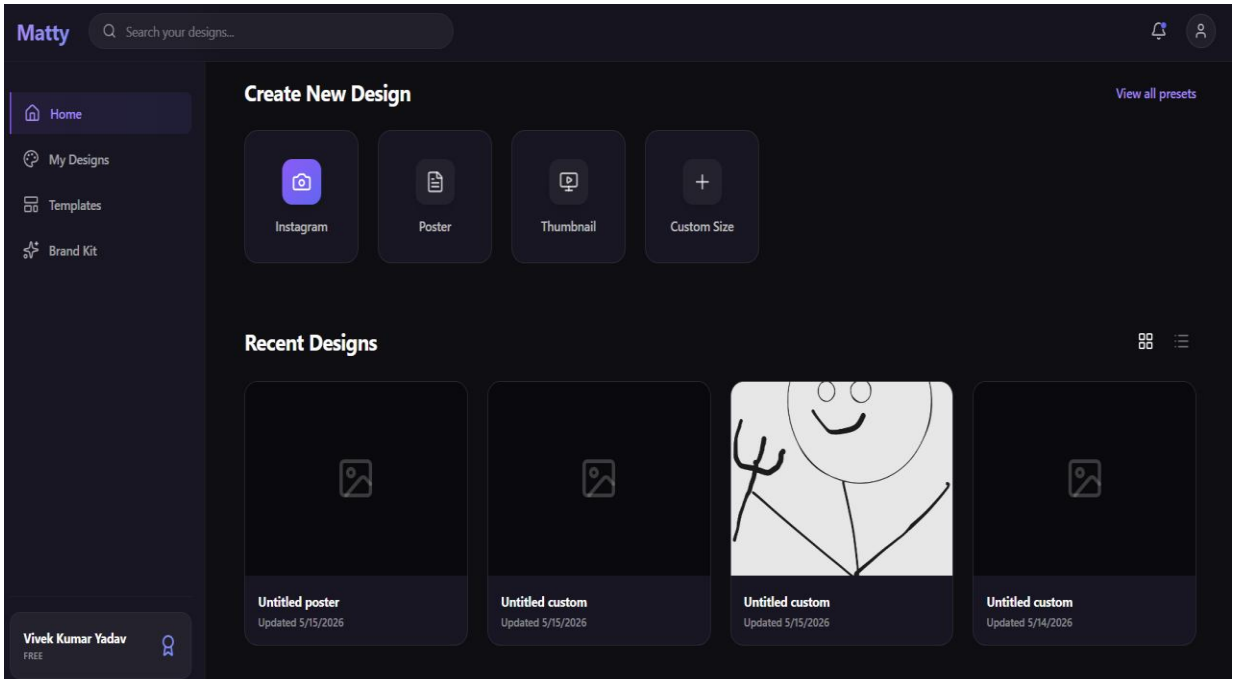
6.2 Project Repositories

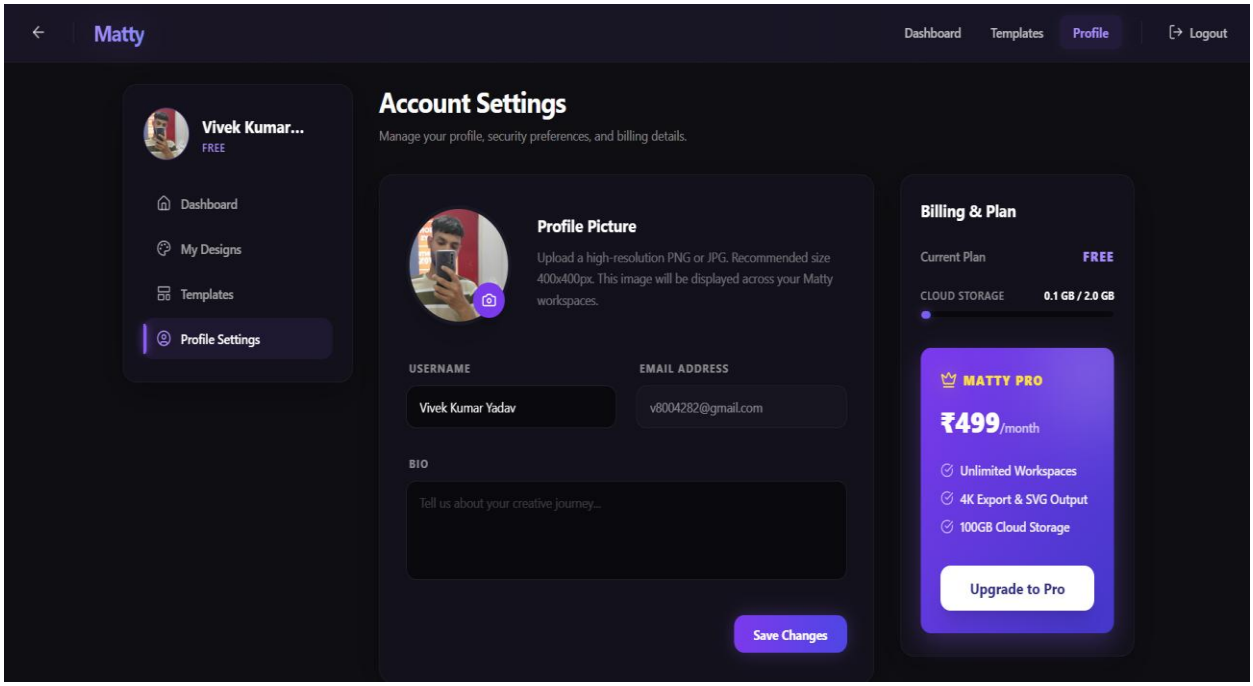
Project	GitHub Repository
Week 1 — Portfolio Website	github.com/vivekYadav129/portfolio
Week 2 — Todo Application	github.com/vivekYadav129/todo-app
Week 3 — React Project Lab	github.com/vivekYadav129/react-project
Week 4 — Blog API	github.com/vivekYadav129/product-api
Week 5 — Task Manager API	github.com/vivekYadav129/task-manager
Capstone — Matty	github.com/vivekYadav129/Matty

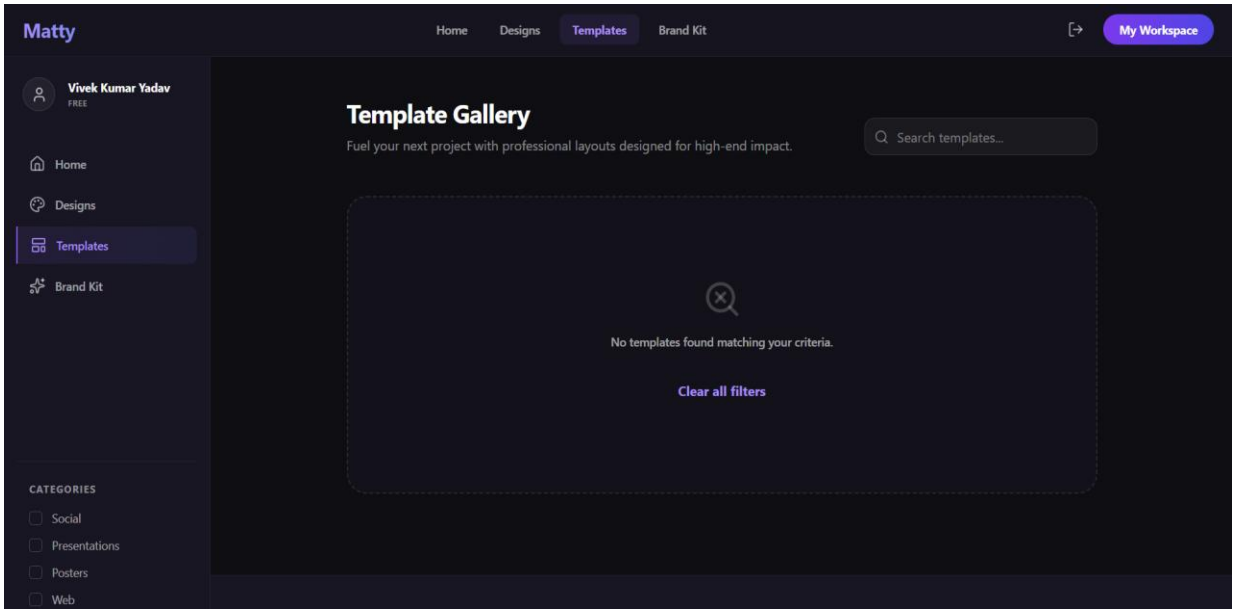
CHAPTER 7: APPENDICES

7.1 SNAPSHOTS






Matty
Dashboard
Templates
Profile
Logout
Vivek Kumar...
FREE
Dashboard
My Designs
Templates
Profile Settings
Account Settings
Manage your profile, security preferences, and billing details.
Profile Picture
Upload a high-resolution PNG or JPG. Recommended size 400x400px. This image will be displayed across your Matty workspaces.
USERNAME
Vivek Kumar Yadav
EMAIL ADDRESS
v8004282@gmail.com
BIO
Tell us about your creative journey...
Save Changes
Billing & Plan
Current Plan
FREE
CLOUD STORAGE
0.1 GB / 2.0 GB
MATTY PRO
₹499/month
Unlimited Workspaces
4K Export & SVG Output
100GB Cloud Storage
Upgrade to Pro


Matty
Home
Designs
Templates
Brand Kit
My Workspace
Vivek Kumar Yadav
FREE
Home
Designs
Templates
Brand Kit
Template Gallery
Fuel your next project with professional layouts designed for high-end impact.
Search templates...
No templates found matching your criteria.
Clear all filters
CATEGORIES
Social
Presentations
Posters
Web