Aim

To deploy a full-stack Task Manager application from GitHub to Vercel and implement automated CI/CD for continuous integration and deployment using GitHub Actions.

Theory

1. Git and GitHub

- **Git** is a distributed version control system that tracks changes in source code during software development.
- **GitHub** is a cloud-based platform built on Git that allows multiple developers to collaborate, store, and manage code repositories.
- Key operations:
    - git add → stages changes.
    - git commit → saves snapshots of the code.
    - git push → uploads local commits to GitHub.
    - git pull → fetches updates from the remote repository.
- Benefits: Version history, collaboration, rollback capabilities, and integration with CI/CD pipelines.

2. Vercel Deployment

- **Vercel** is a cloud platform for deploying web applications, including static websites, React, Next.js, and Node.js apps.
- Features:
    - **Automatic Builds:** Vercel detects the project type and runs build commands automatically.

    - **Instant Deployments:** Provides a live URL immediately after deployment.
    - **Environment Variables:** Allows secure configuration for sensitive data like database URLs and API keys.
- Advantages:
    - No manual server setup.
    - Scalable deployments.
    - Seamless integration with GitHub for continuous deployment.

3. Continuous Integration (CI)

- CI is a development practice where developers **frequently merge code changes** into a central repository.
- Automated processes run **tests, linting, and builds** to ensure code quality and prevent integration issues.
- Tools: GitHub Actions, Jenkins, Travis CI, CircleCI.

4. Continuous Deployment (CD)

- CD automates the **release of code to production** once it passes the CI pipeline.
- Eliminates manual deployment errors and ensures the live application is always up-to-date.
- Vercel acts as the CD platform in this experiment: every push to GitHub triggers a new build and deployment automatically.

5. GitHub Actions

- GitHub Actions is a **workflow automation tool** integrated into GitHub repositories.
- Workflow is defined in .github/workflows/*.yml files.
- Steps in a typical workflow:

1. Checkout code.
2. Set up runtime (Node.js, Python, etc.).
3. Install dependencies.
4. Run tests.
5. Build project.
6. Deploy to cloud platform (like Vercel).

- Benefits: Fully automated CI/CD, integrated with GitHub, reduces manual intervention.

6. Full-stack Deployment Workflow

1. **Clone the repository** locally using git clone.
2. **Install dependencies** using package managers like npm.
3. **Push the project** to GitHub repository.
4. **Connect GitHub repo to Vercel** for deployment.
5. **Configure environment variables** (JWT secret, database URL, etc.).
6. **Automated CI/CD** triggers on code push:
    - Builds the application.
    - Runs tests.
    - Deploys to Vercel automatically.
7. **Access the live application** via the Vercel-provided URL.

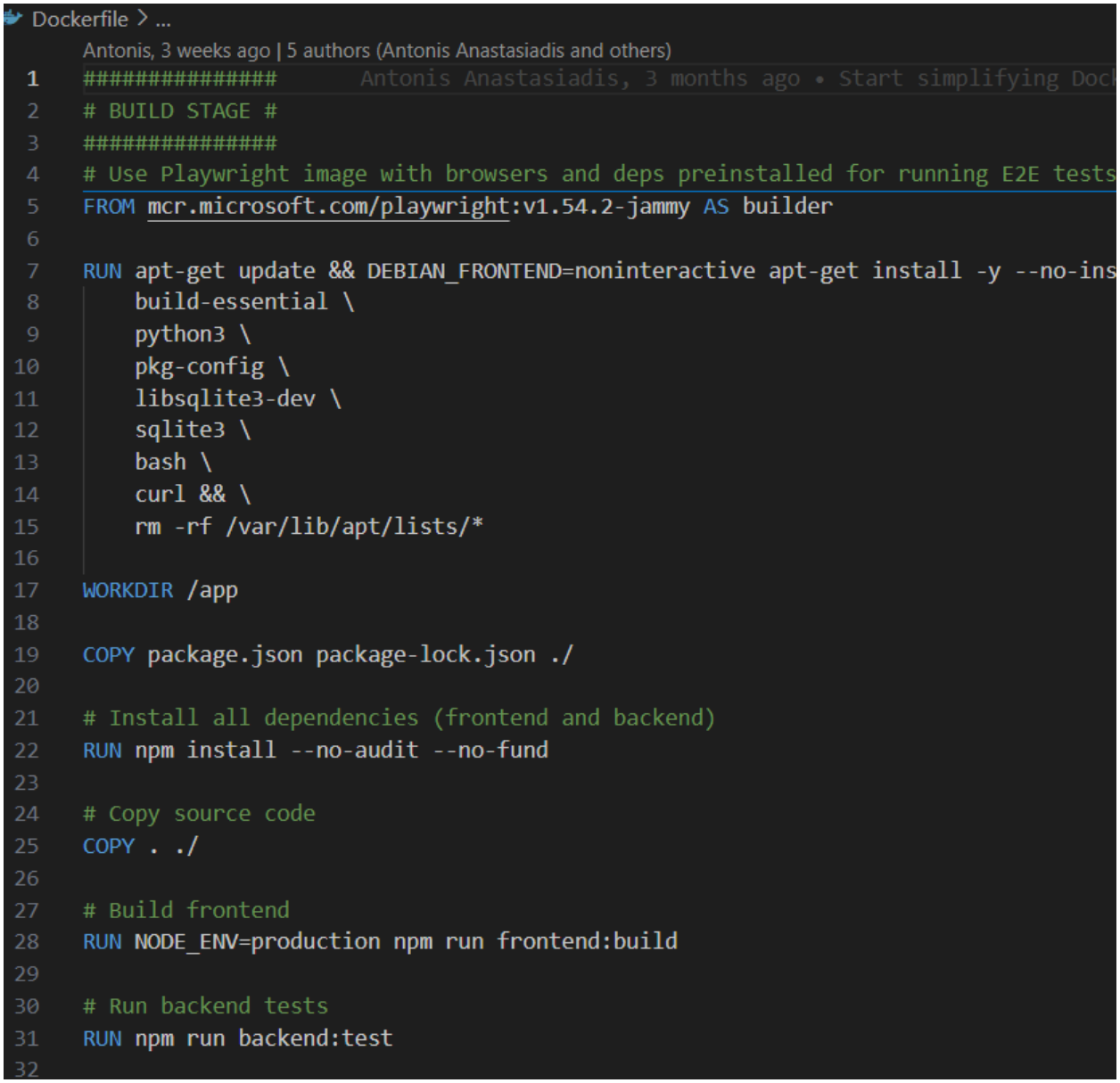
7. Advantages of this Setup

- Automation reduces human error.
- Live application always reflects the latest code.

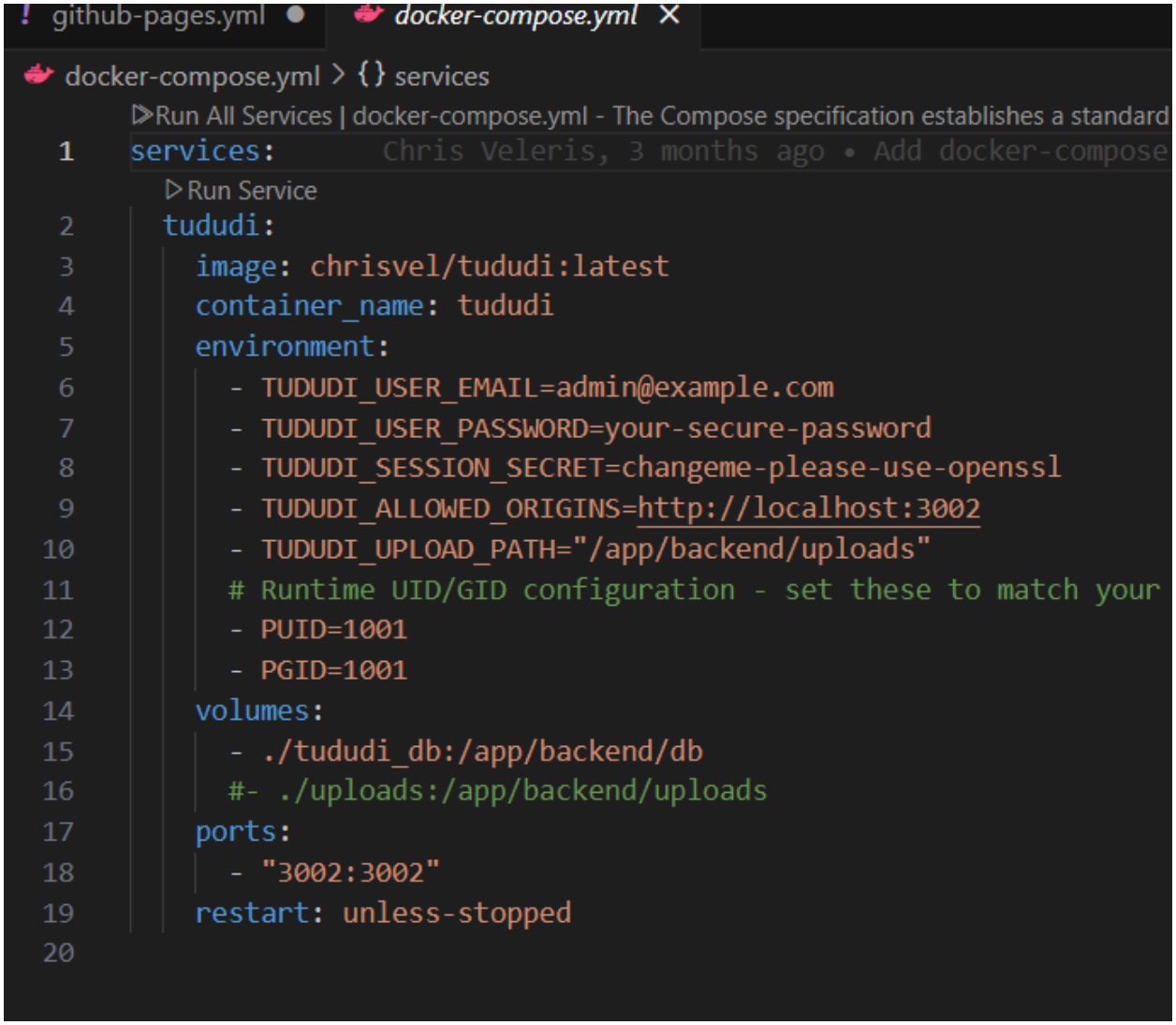
- Easy collaboration and code review through GitHub.
- Rapid deployment and rollback capabilities.
- Modern software development practice aligned with industry standards.

CI/CD WORKFLOW- .yml file

Dockerfile


Dockerfile > ...
Antonis, 3 weeks ago | 5 authors (Antonis Anastasiadis and others)
1 ################ Antonis Anastasiadis, 3 months ago • Start simplifying Doc
2 # BUILD STAGE #
3 ################
4 # Use Playwright image with browsers and deps preinstalled for running E2E tests
5 FROM mcr.microsoft.com/playwright:v1.54.2-jammy AS builder
6
7 RUN apt-get update && DEBIAN_FRONTEND=noninteractive apt-get install -y --no-ins
8 build-essential \
9 python3 \
10 pkg-config \
11 libsqlite3-dev \
12 sqlite3 \
13 bash \
14 curl && \
15 rm -rf /var/lib/apt/lists/*
16
17 WORKDIR /app
18
19 COPY package.json package-lock.json ./
20
21 # Install all dependencies (frontend and backend)
22 RUN npm install --no-audit --no-fund
23
24 # Copy source code
25 COPY . ./
26
27 # Build frontend
28 RUN NODE_ENV=production npm run frontend:build
29
30 # Run backend tests
31 RUN npm run backend:test
32

Docker-compose.yml



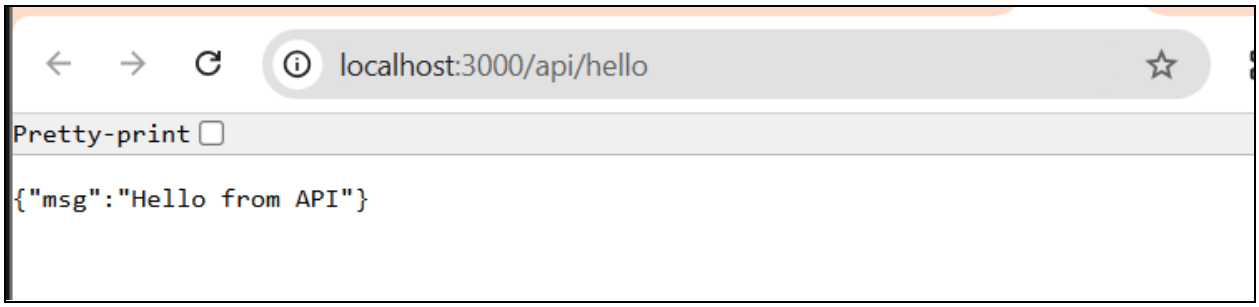
Index.tsx

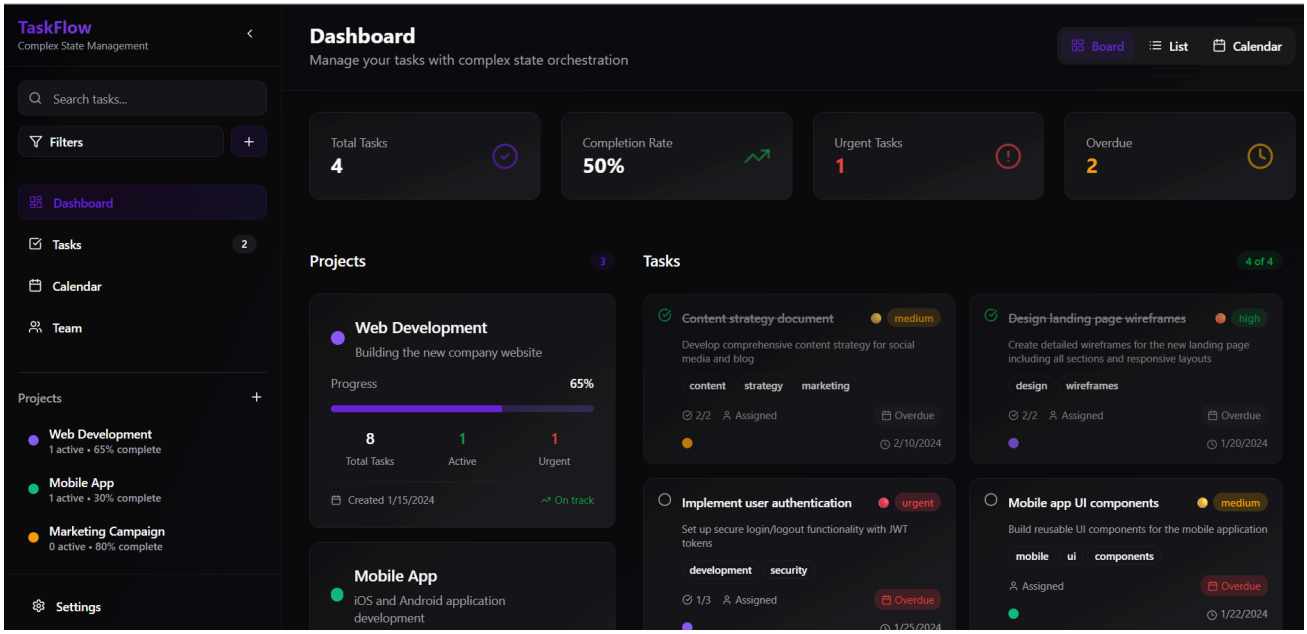
```tsx
};

import React from 'react';
import { createRoot } from 'react-dom/client';
import { BrowserRouter } from 'react-router-dom';
import App from './App';
import { ToastProvider } from './components/Shared/ToastContext';
import { TelegramStatusProvider } from './contexts/TelegramStatusContext';
import './i18n'; // Import i18n config to initialize it
import './styles/markdown.css'; // Import markdown styles
import { I18nextProvider } from 'react-i18next';
import i18n from './i18n'; // Import the i18n instance with its configuration

const storedPreference = localStorage.getItem('isDarkMode');
const prefersDarkMode = window.matchMedia(
    '(prefers-color-scheme: dark)'
).matches;
const isDarkMode = storedPreference
    ? storedPreference === 'true'
    : prefersDarkMode;

if (isDarkMode) {
    document.documentElement.classList.add('dark');
} else {
    document.documentElement.classList.remove('dark');
}

const container = document.getElementById('root');

// Store the root outside the if block so it can be accessed by the HMR code
let root: any;

if (container) {
```

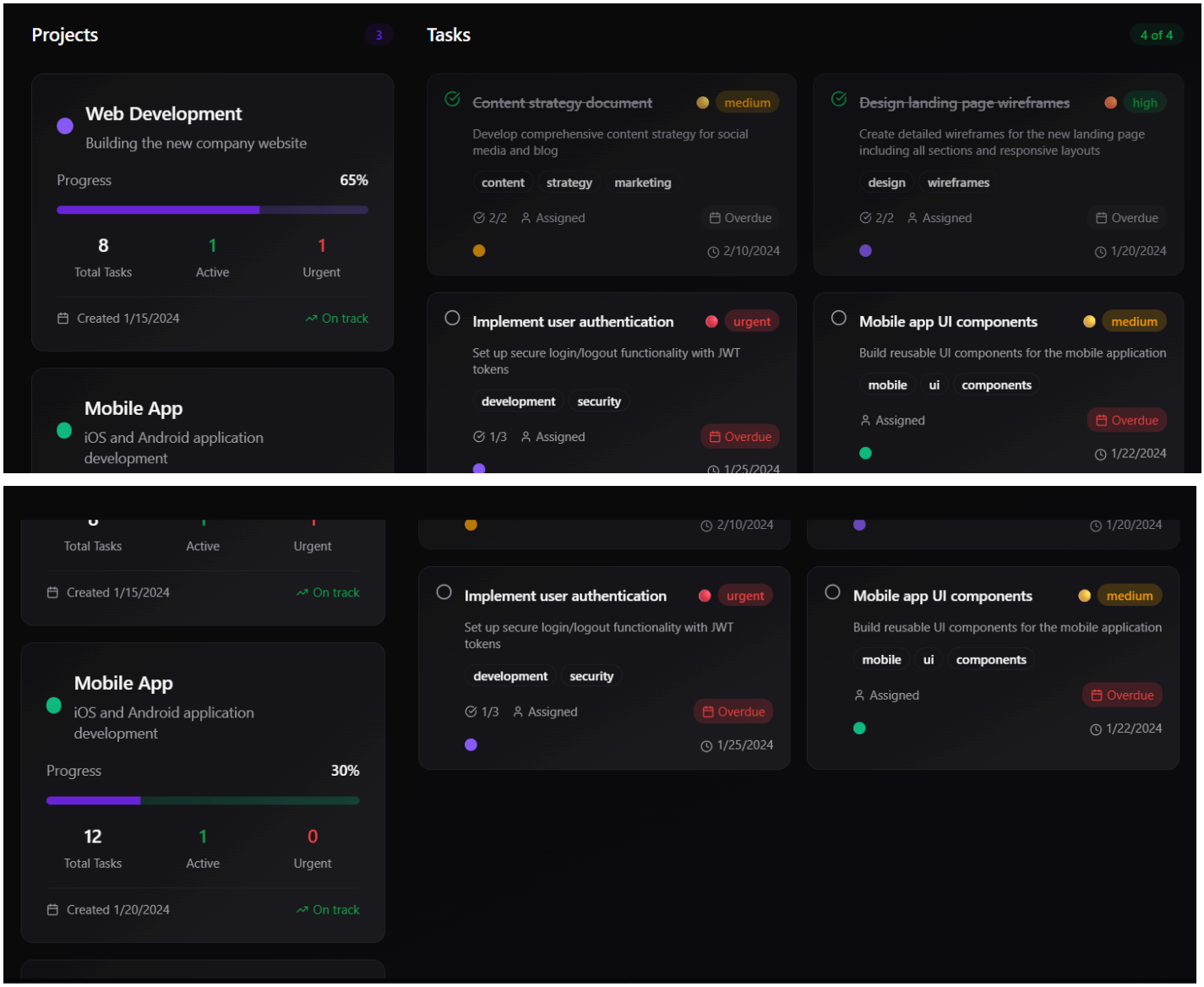
App.tsx

```tsx
import TaskDetails from './components/Task/TaskDetails';
import LoadingScreen from './components/Shared/LoadingScreen';
import InboxItems from './components/Inbox/InboxItems';
// Lazy load Tasks component to prevent issues with tags loading
const Tasks = lazy(() => import('./components/Tasks'));

const App: React.FC = () => {
    const { i18n } = useTranslation();
    const [currentUser, setCurrentUser] = useState<User | null>(null);
    const [loading, setLoading] = useState(true);

    if (!i18n.isInitialized) {
        return <LoadingScreen />;
    }

    const fetchCurrentUser = async () => {
        try {
            const response = await fetch('/api/current_user', {
                credentials: 'include',
                headers: {
                    Accept: 'application/json',
                },
            });

            if (!response.ok) {
                if (response.status === 401) {
                    setCurrentUser(null);
                    return;
                }
                throw new Error(`Failed to fetch user: ${response.status}`);
            }

            const data = await response.json();
```

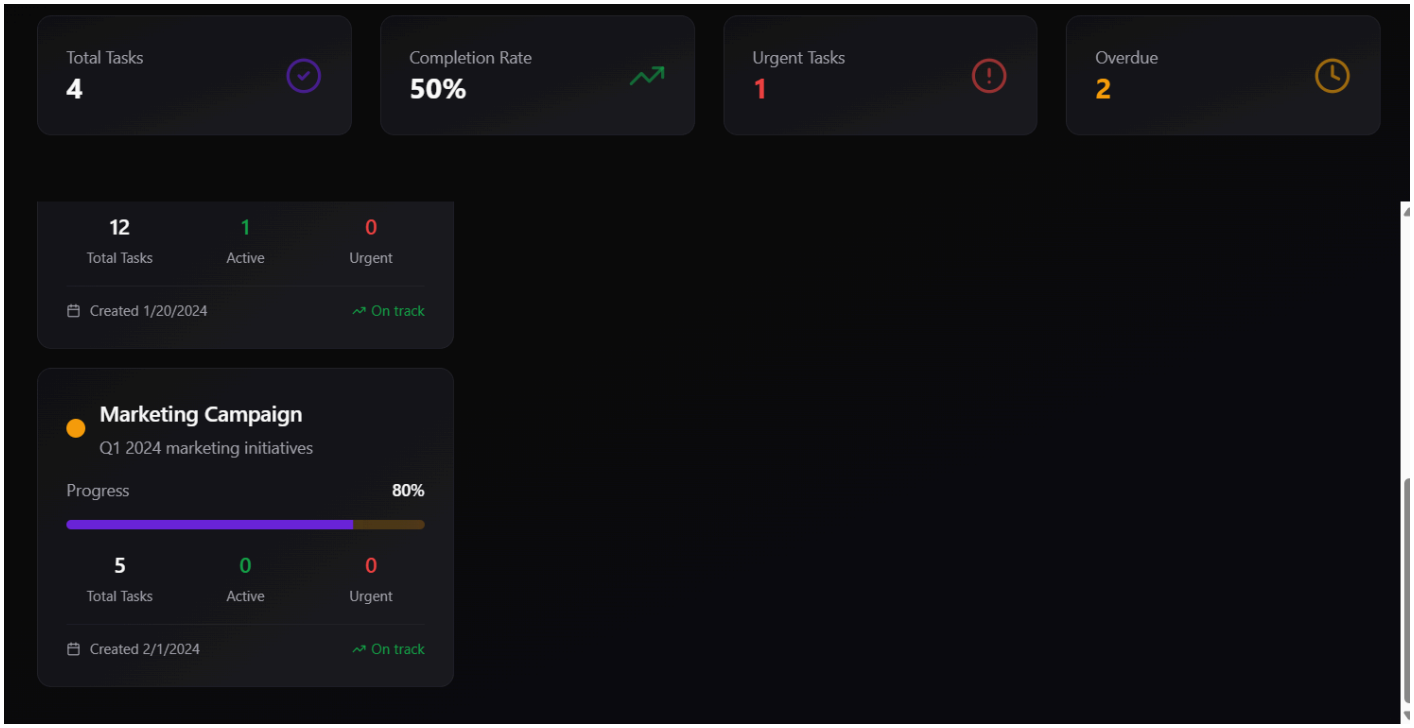
```
roject\exp10\backend> npm run dev

> backend@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
[dotenv@17.2.3] injecting env (0) from .env -- tip: 🔐 encrypt with Dotenvx: https://dotenvx.com
API listening on 3000
curl http://localhost:3000/api/hello
```

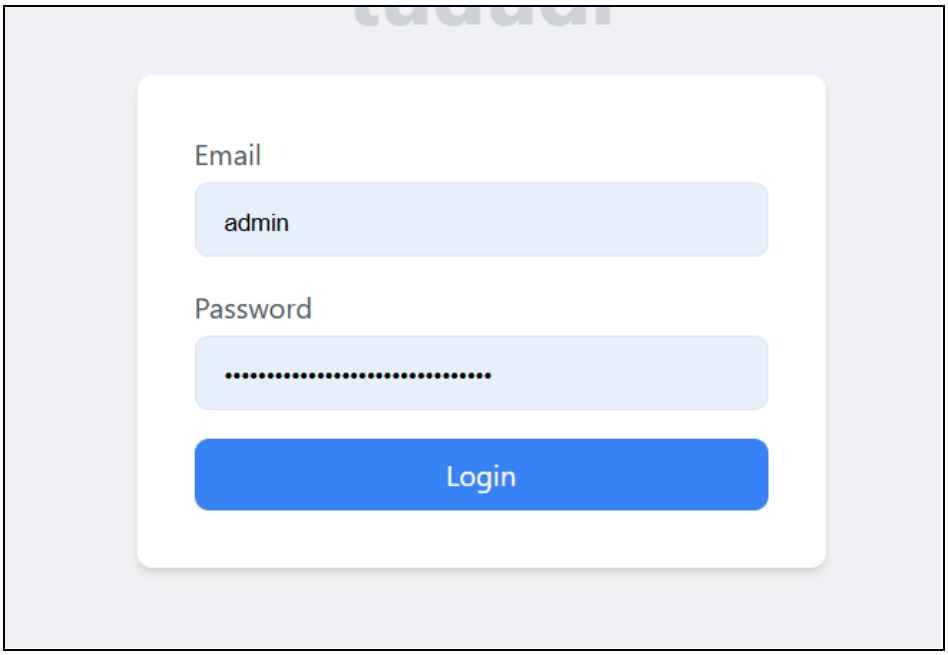

localhost:3000/api/hello
Pretty-print
{"msg":"Hello from API"}

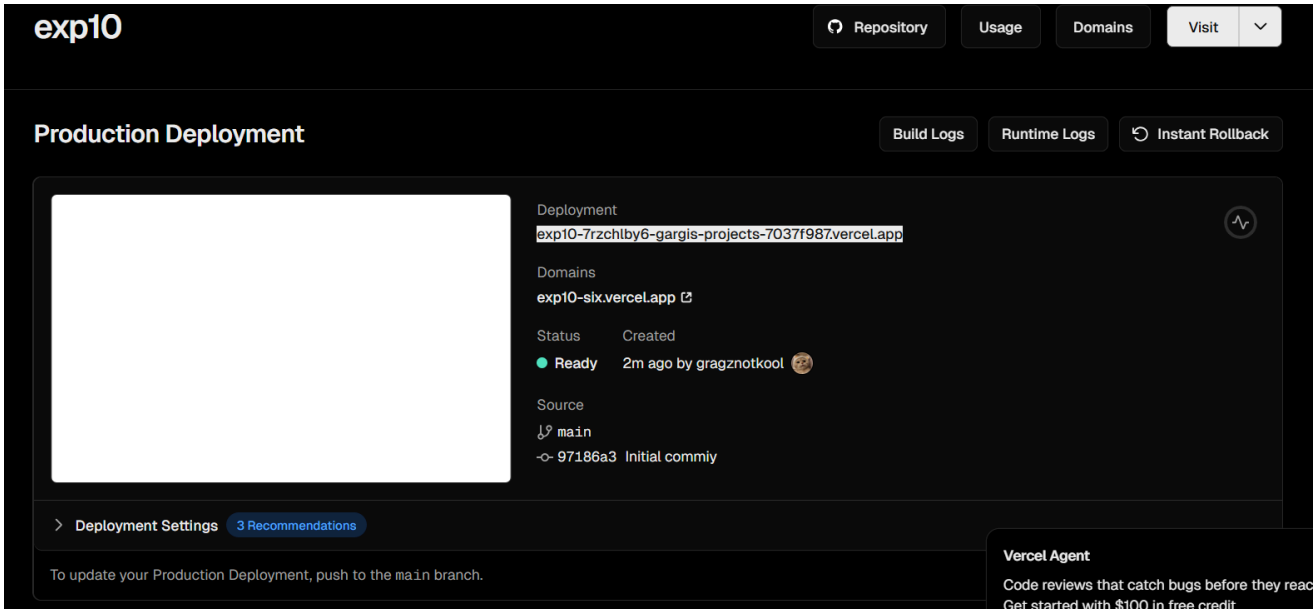

TaskFlow
Complex State Management
Search tasks...
Filters
Dashboard
Tasks 2
Calendar
Team
Projects
Web Development
1 active • 65% complete
Mobile App
1 active • 30% complete
Marketing Campaign
0 active • 80% complete
Settings
Dashboard
Manage your tasks with complex state orchestration
Board
List
Calendar
Total Tasks
4
Completion Rate
50%
Urgent Tasks
1
Overdue
2
Projects
3
Web Development
Building the new company website
Progress
65%
8
Total Tasks
1
Active
1
Urgent
Created 1/15/2024
On track
Mobile App
iOS and Android application development
Tasks
4 of 4
Content strategy document
medium
Develop comprehensive content strategy for social media and blog
content
strategy
marketing
2/2
Assigned
Overdue
2/10/2024
Design landing page wireframes
high
Create detailed wireframes for the new landing page including all sections and responsive layouts
design
wireframes
2/2
Assigned
Overdue
1/20/2024
Implement user authentication
urgent
Set up secure login/logout functionality with JWT tokens
development
security
1/3
Assigned
Overdue
Mobile app UI components
medium
Build reusable UI components for the mobile application
mobile
ui
components
Assigned
Overdue
1/22/2024


Projects
3
Web Development
Building the new company website
Progress
65%
8
Total Tasks
1
Active
1
Urgent
Created 1/15/2024
On track
Mobile App
iOS and Android application development
Tasks
4 of 4
Content strategy document
medium
Develop comprehensive content strategy for social media and blog
content
strategy
marketing
2/2
Assigned
Overdue
2/10/2024
Design landing page wireframes
high
Create detailed wireframes for the new landing page including all sections and responsive layouts
design
wireframes
2/2
Assigned
Overdue
1/20/2024
Implement user authentication
urgent
Set up secure login/logout functionality with JWT tokens
development
security
1/3
Assigned
Overdue
1/25/2024
Mobile app UI components
medium
Build reusable UI components for the mobile application
mobile
ui
components
Assigned
Overdue
1/22/2024

Total Tasks
Active
Urgent
Created 1/15/2024
On track
Mobile App
iOS and Android application development
Progress
30%
12
Total Tasks
1
Active
0
Urgent
Created 1/20/2024
On track
2/10/2024
1/20/2024
Implement user authentication
urgent
Set up secure login/logout functionality with JWT tokens
development
security
1/3
Assigned
Overdue
1/25/2024
Mobile app UI components
medium
Build reusable UI components for the mobile application
mobile
ui
components
Assigned
Overdue
1/22/2024


Total Tasks
4
Completion Rate
50%
Urgent Tasks
1
Overdue
2
12
Total Tasks
1
Active
0
Urgent
Created 1/20/2024
On track
Marketing Campaign
Q1 2024 marketing initiatives
Progress
80%
5
Total Tasks
0
Active
0
Urgent
Created 2/1/2024
On track





https://exp10-7rzchlby6-gargis-projects-7037f987.vercel.app/

Conclusion

- The Task Manager project was successfully deployed to Vercel from GitHub.
- Continuous deployment ensures that any changes pushed to GitHub automatically reflect on the live application.
- CI/CD integration improves development efficiency, reduces human error, and allows real-time updates of the application.
- This experiment demonstrates modern cloud deployment practices and the importance of version control, automated testing, and continuous delivery in software development.