Frontend

in app.ts

```
import axios from 'axios';

// Base API URL
const API_URL = 'http://localhost:3000'; // Replace with actual API URL
      rawan mahmoud, 3 days ago • frontend w backend edits  rbna yestor
```

```
      You, 2 days ago • estor yard
// Course Interface
You, 12 hours ago | 2 authors (You and one other)
export interface Course {
  courseId: string;
  title: string;
  description: string;
  instructorId: string;
  role: string;
  version: number;
  status: string;
  modules: string[];
}
```

```
// 3. Get course details by ID
export const getCourseDetails = async (id: string): Promise<Course> => {
  const response = await axios.get(`${API_URL}/courses/Cbyid/${id}`);
  return response.data;
};      You, 12 hours ago • Uncommitted changes
```

for login

```
// Login Credentials
You, 12 hours ago | 2 authors (You and one other)
export interface Credentials {
  email: string;
  password: string;
}

// Login Response
You, 12 hours ago | 2 authors (You and one other)
export interface LoginResponse {
  accessToken: string;
  role: string;
  _id: string;
  name: string;
}
```

```
// 1. Login
export const login = async (data: Credentials): Promise<LoginResponse> =>
  const response = await axios.post(`${API_URL}/users/login`, data);
  return response.data;
};
```

CourcePage.tsx

```tsx
"use client";

import React, { useEffect, useState } from "react";
import { getCourses, Course } from "../utils/api";
import CourseCard from "../components/CourseCard";

const CoursesPage: React.FC = () => {
  const [courses, setCourses] = useState<Course[]>([]);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState<string | null>(null);
```

```tsx
  // Fetch courses when the component loads
  useEffect(() => {
    const fetchCourses = async () => {
      try {
        const fetchedCourses = await getCourses();
        setCourses(fetchedCourses);
      } catch (err) {
        console.error("Failed to fetch courses:", err);
        setError("Failed to load courses. Please try again later.");
      } finally {
        setLoading(false);
      }
    };

    fetchCourses();
  }, []);
```

```
  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold text-center mb-4">Courses</h1>
      {courses.length ? (
        <div className="grid gap-4 grid-cols-1 md:grid-cols-2 lg:grid-cols-3">
          {courses.map((course) => (
            <CourseCard key={course.courseId} course={course} />
          ))}
        </div>
      ) : (
        <p className="text-center mt-10">No courses available.</p>
      )}
    </div>
  );
};

export default CoursesPage;
```

CourseCard.tsx

tsx

```
import React from "react";

interface CourseCardProps {
  course: {
    title: string;
    description: string;
    version: number;
    status: string;
  };
}
```

```tsx
const CourseCard: React.FC<CourseCardProps> = ({ course }) => (
  <div className="bg-white shadow-md rounded p-4">
    <h2 className="text-xl font-bold text-blue-600">{course.title}</h2>
    <p className="text-gray-700">{course.description}</p>
    <p className="text-gray-500 text-sm">Version: {course.version}</p>
    <p className={course.status === "valid" ? "text-green-500" : "text-red-500"}>
      Status: {course.status[0].toUpperCase() + course.status.slice(1)}
    </p>
  </div>
);

export default CourseCard;
```