TechEdu

Digital Egypt Pioneers Initiative

Final Project

Team Names

1) Ziad Essam Abdelaziz AbdelMoula
2) Karim Hazem Hamed Mohamed

Under The Supervision of

Eng.Rowida Adel

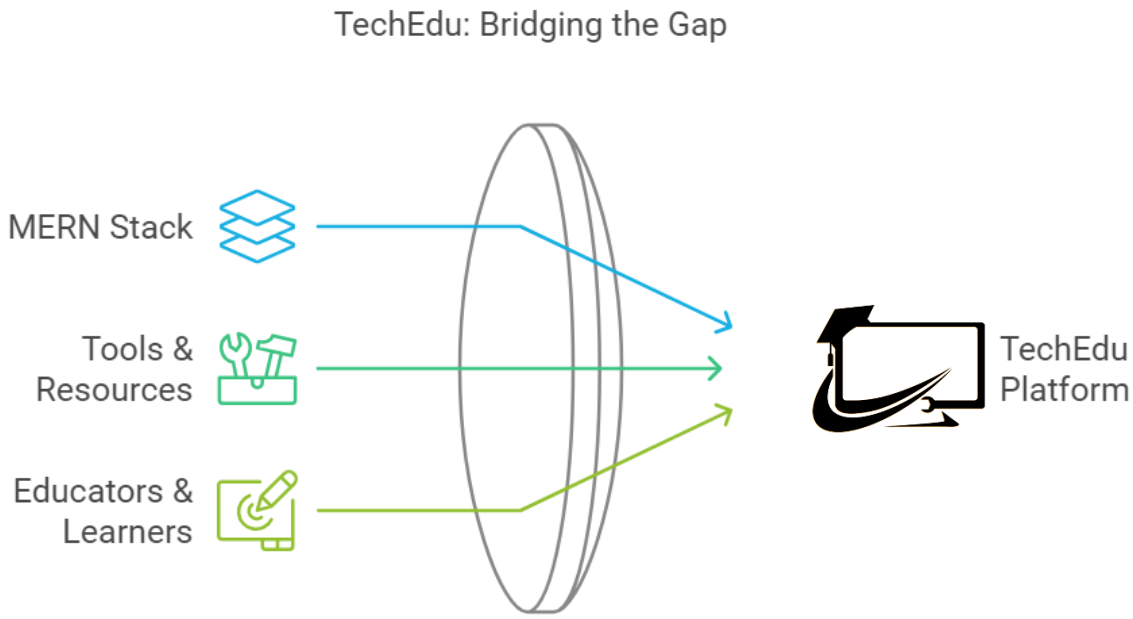
October 2024

Table Of Content

Chapter 1: Introduction

1.1 Project Overview

TechEdu is an online education platform designed to facilitate learning and teaching across a variety of disciplines. It leverages the MERN stack (MongoDB, Express.js, React.js, Node.js) to deliver a seamless and interactive user experience. The platform aims to bridge the gap between educators and learners by providing a comprehensive suite of tools and resources.



1.2 Objectives

The primary objectives of TechEdu are:

- **To provide accessible and high-quality education to a global audience.**
- **To offer a robust platform for educators to create and manage their courses.**
- **To enable students to engage in interactive learning experiences.**
- **To integrate advanced technologies to enhance the learning process.**

1.3 Scope

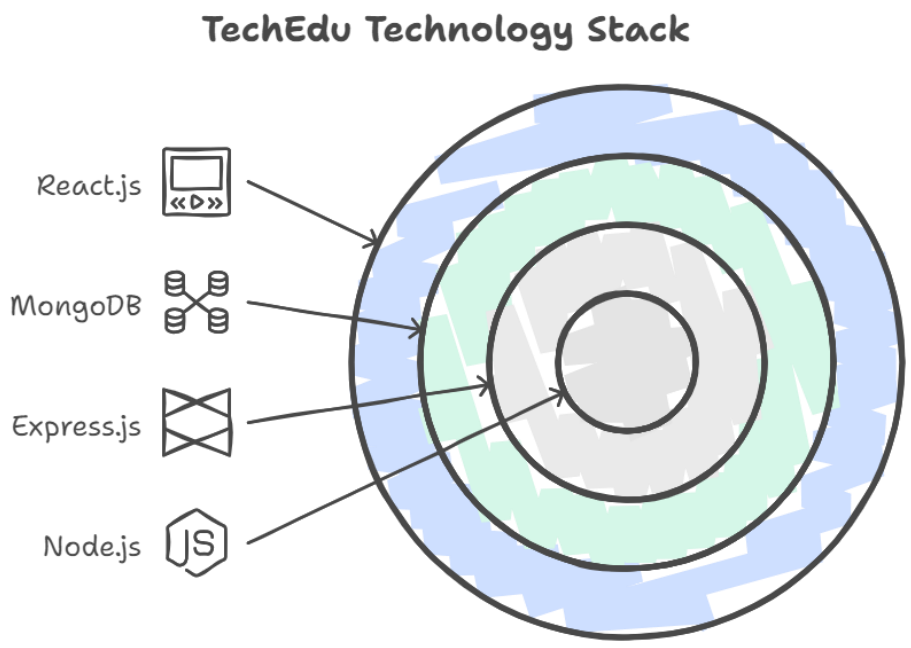
The scope of TechEdu includes:

- **Development of a user-friendly front end for learners using vite and react**
- **Creation of an admin portal for educators and administrators using React.**
- **Implementation of a backend API using Node.js and Express to handle data operations.**
- **Integration with MongoDB to manage and store course-related data.**
- **Deployment and maintenance of the platform.**

1.4 Technologies Used

TechEdu is built using the following technologies:

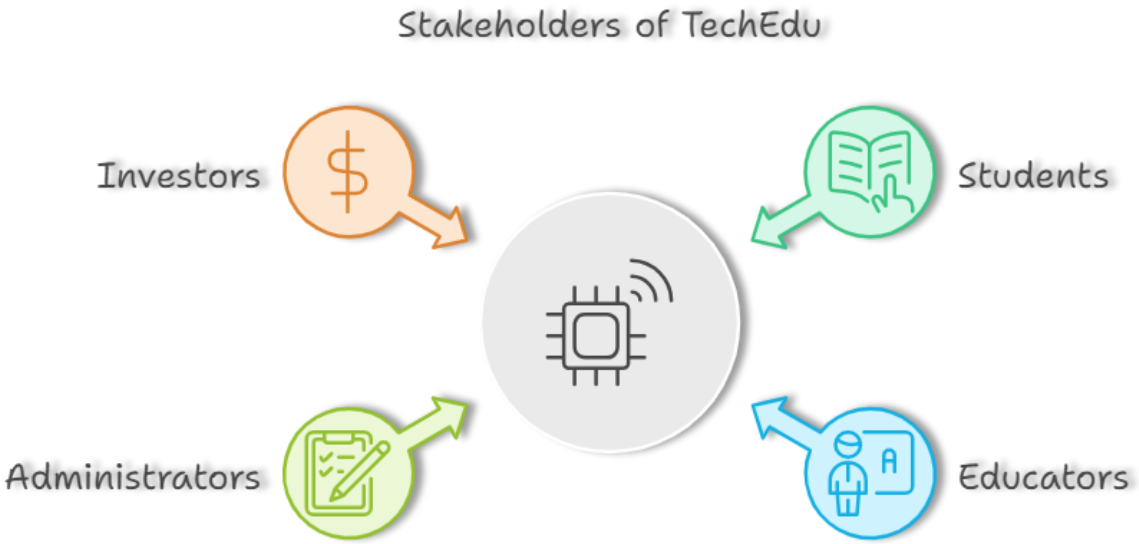
- **MongoDB: For database management.**
- **Express.js: For backend framework.**
- **React.js: For frontend development.**
- **Node.js: For server-side runtime.**



1.5 Audience and Stakeholders

The primary audience and stakeholders of TechEdu are:

- **Students: Individuals seeking to learn new skills and knowledge.**
- **Educators: Professionals creating and managing educational content.**
- **Administrators: Users overseeing the platform's operations and maintenance.**
- **Investors: Parties interested in the development and success of the platform.**



Chapter 2: Planning

2.1 Project Plan

The project plan outlines the roadmap for developing TechEdu. It includes the objectives, milestones, and the timeline needed to complete the project successfully.

2.2 Timeline

Phase 1: Gathering and Analysis Requirements (1 week)

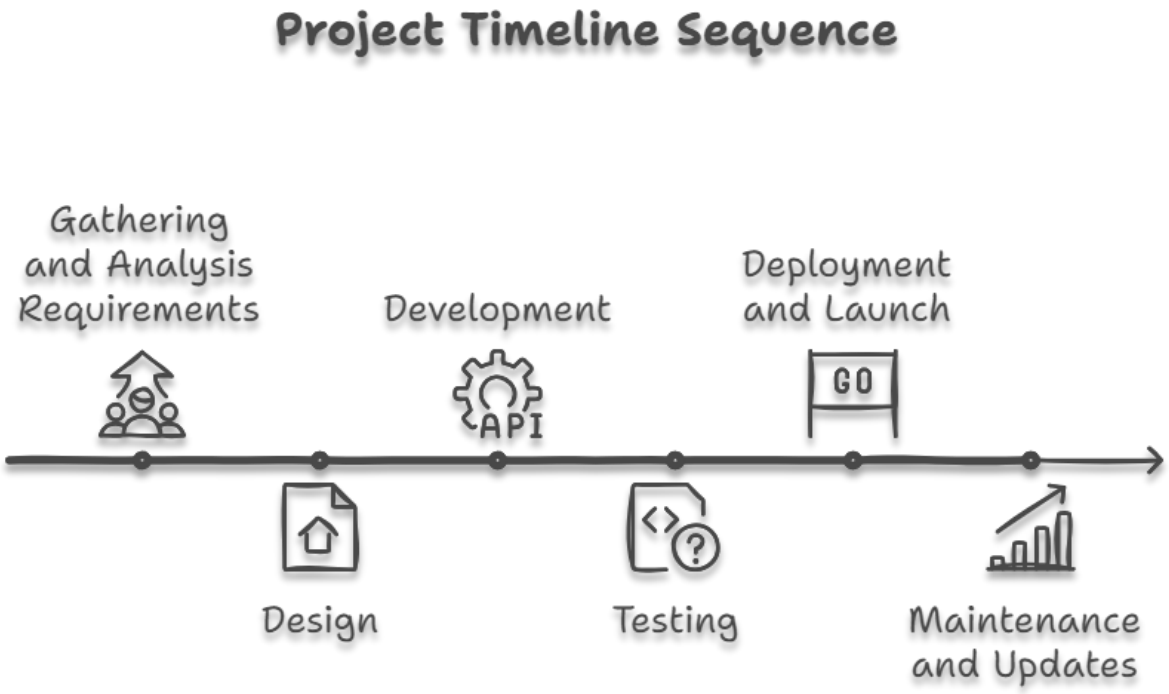
Phase 2: Design (1 weeks)

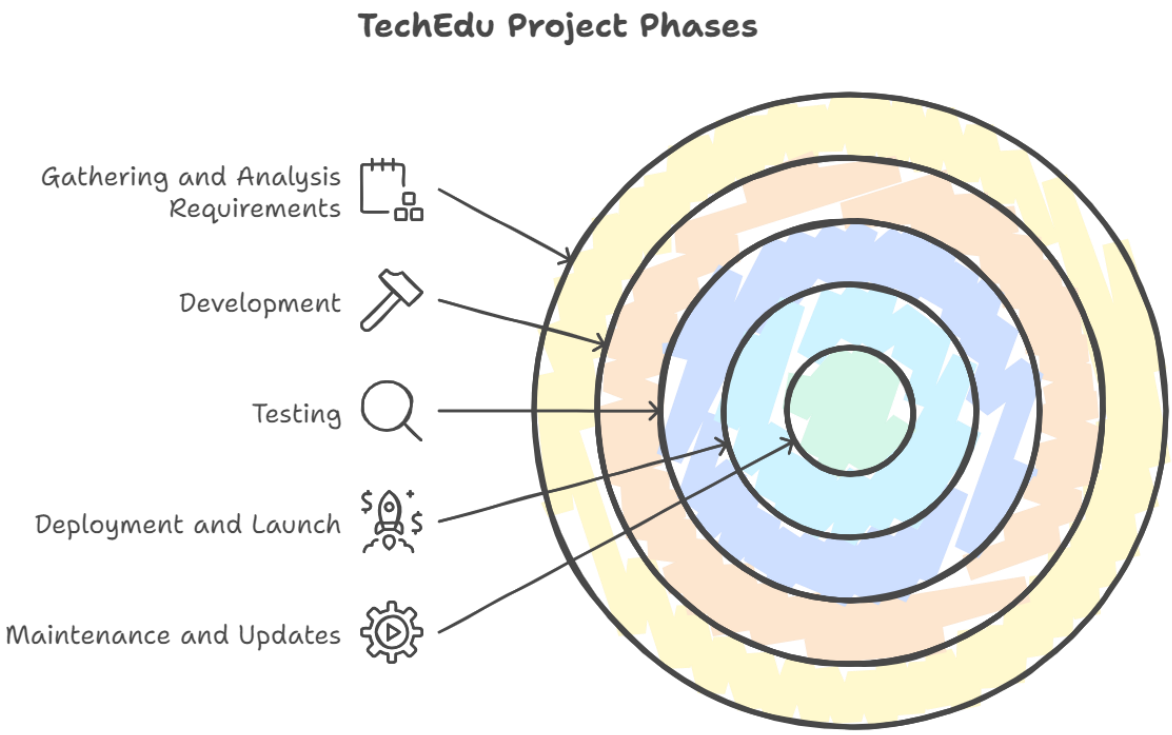
Phase 3: Development (2 weeks)

Phase 4: Testing (4 weeks)

Phase 5: Deployment and Launch (2 weeks)

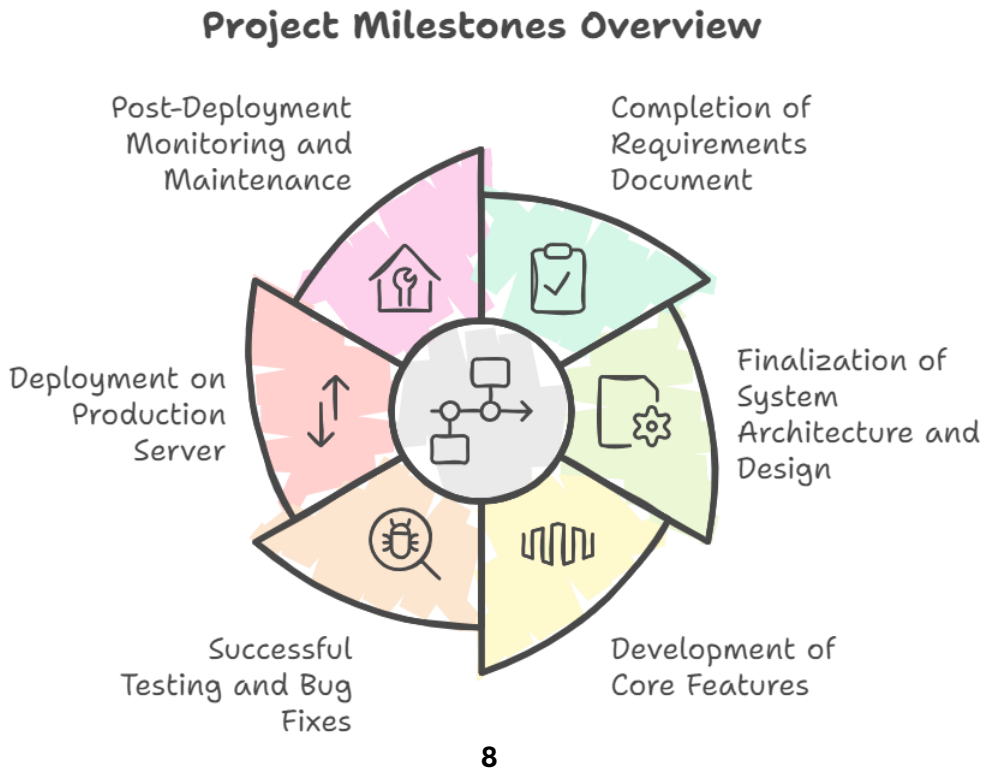
Phase 6: Maintenance and Updates (Ongoing)





2.3 Milestones

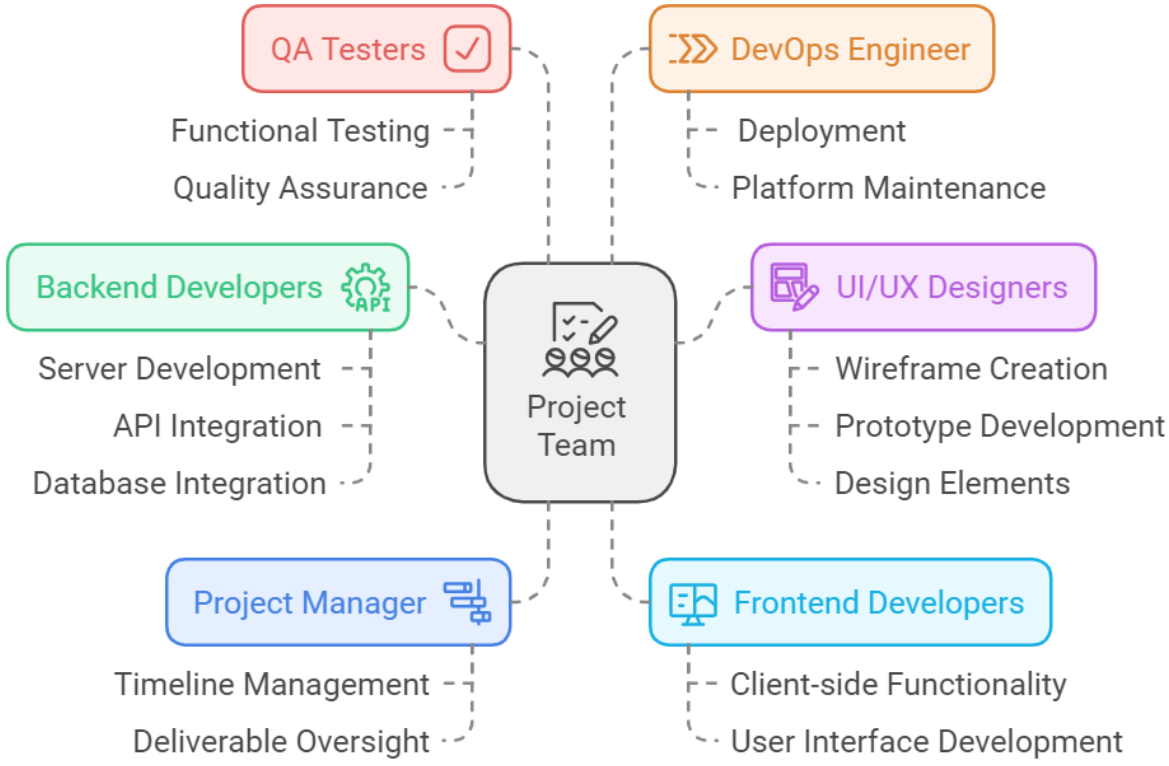
- **M1: Completion of Requirements Document**
- **M2: Finalization of System Architecture and Design**
- **M3: Development of Core Features (Frontend and Backend)**
- **M4: Successful Testing and Bug Fixes**
- **M5: Deployment of Production Server**
- **M6: Post-Deployment Monitoring and Maintenance**



2.4 Resource Allocation

Allocate resources effectively to ensure each phase of the project is completed on time. Key resources include:

1. **Project Manager: Overseeing the project timeline and deliverables.**
2. **Frontend Developers: Building the user interface and client-side functionalities.**
3. **Backend Developers: Developing the server, API, and database integration.**
4. **UI/UX Designers: Creating wireframes, prototypes, and design elements.**
5. **QA Testers: Conducting various tests to ensure the quality and functionality of the platform.**
6. **DevOps Engineer: Handling deployment and maintenance of the platform.**



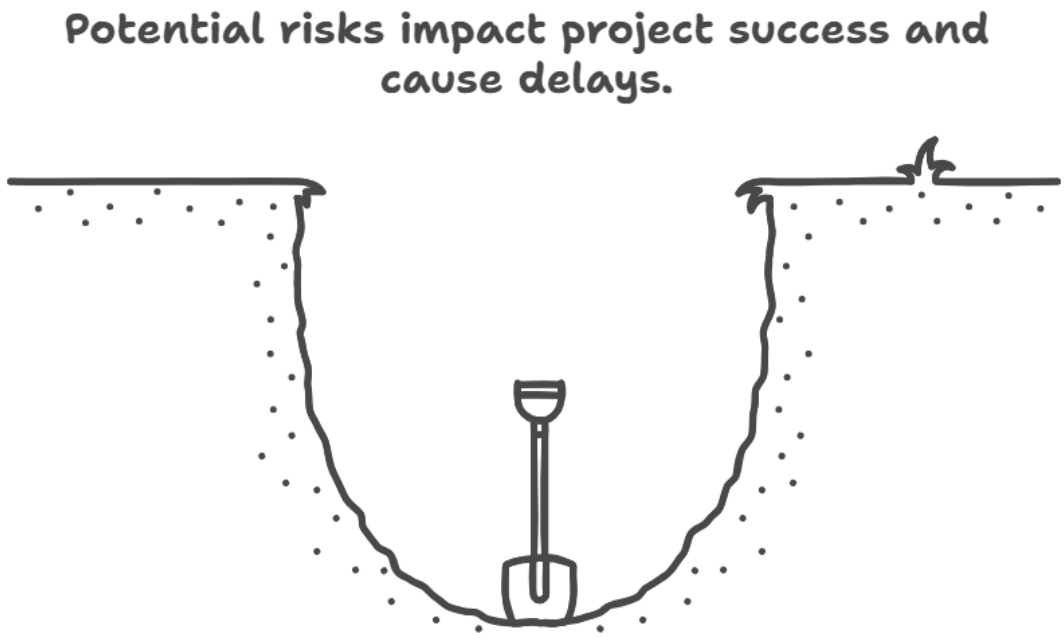
2.5 Risk Management

Identify and manage potential risks that could impact the project. This involves:

Risk Identification: List possible risks (e.g., technical challenges, timeline delays, resource constraints).

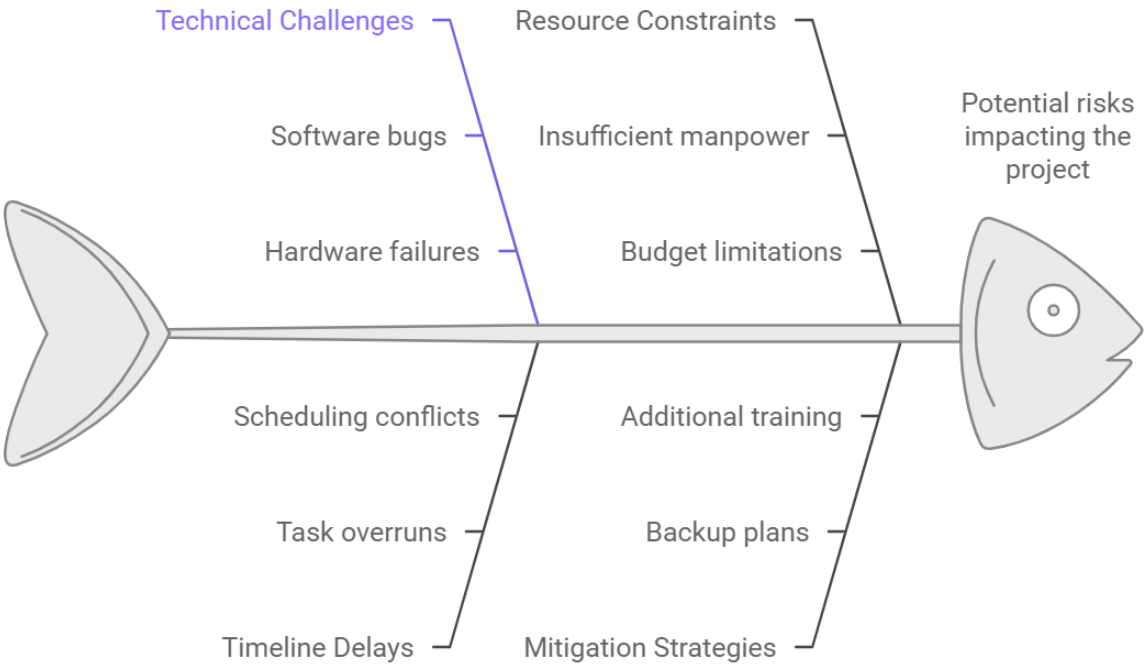
Risk Assessment: Determine the likelihood and impact of each risk.

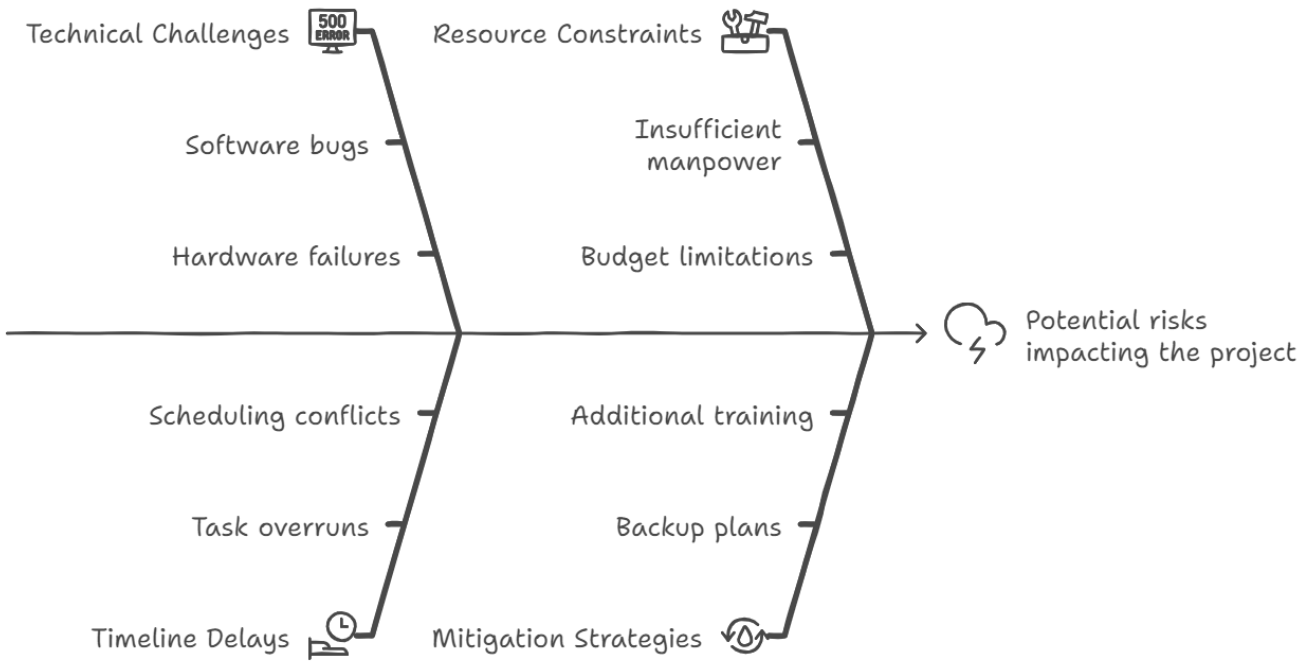
Mitigation Strategies: Develop strategies to mitigate identified risks (e.g., backup plans, additional training, contingency funds).



Let's Go on the Fish Diagram to more details

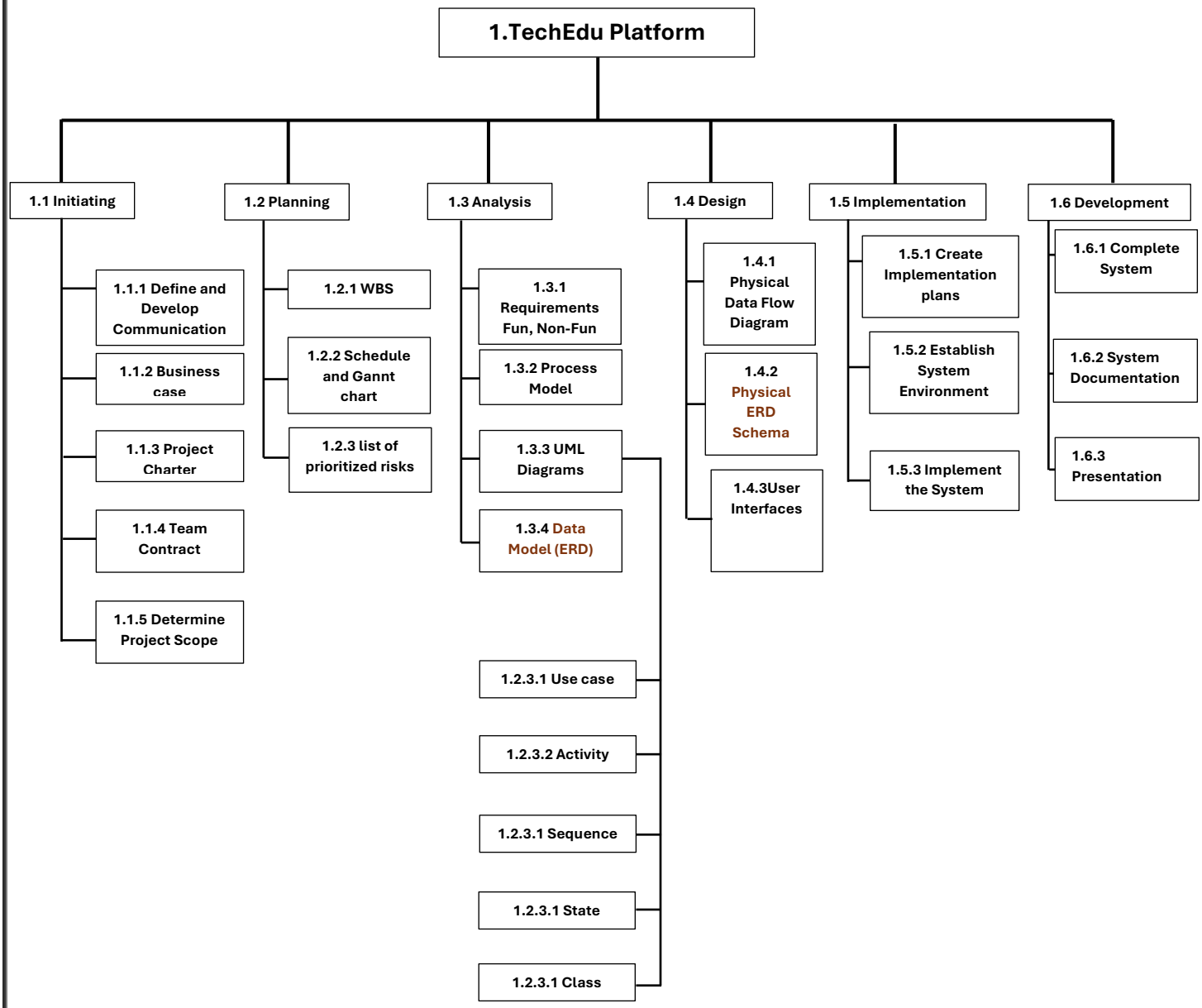
Project Risk Management





2.6 TechEdu Work Break Down Structure (WBS)

Breaking work into smaller tasks is a common productivity technique used to make the work more manageable and approachable for our project TechEdu Website.

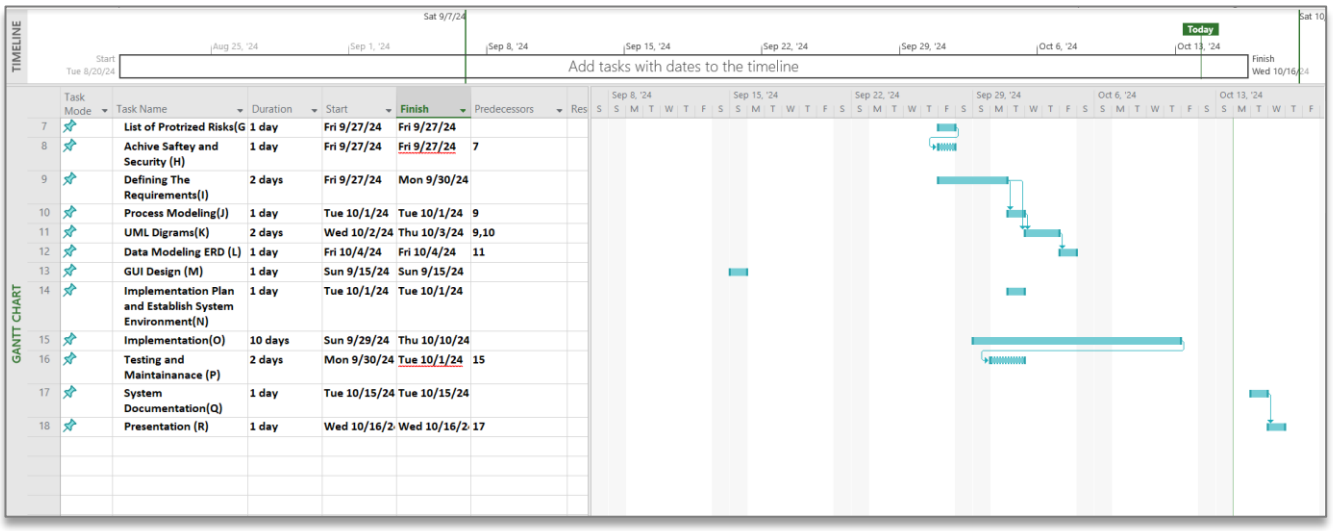


2.7 TechEdu Schedule and Gannt Chart

TechEdu Schedule

Activity	Name	Effort (day)	Start date	End date
Initiating Communication plan and Business case	A	3	15/9/2024	17/9/2024
Project Charter	B	2	18/9/2024	19/9/2024
Team Contract	C	2	20/9/2024	21/9/2024
Determine Project Scope	D	2	22/9/2024	24/9/2024
WBS	E	1	25/9/2024	25/9/2024
Project Schedule	F	1	26/9/2024	26/9/2024
list of Prioritized Risks	G	1	27/9/2024	27/9/2024
How to achieve safety and security	H	1	27/9/2024	27/9/2024
Defining Requirement	I	2	27/9/2024	29/9/2024
Process Modeling	J	1	29/9/2024	29/9/2024
UML Diagrams	K	2	30/9/2024	1/10/2024
Data Modeling ERD	L	1	30/9/2024	1/10/2024
GUI Design	M	1	15/9/2024	16/9/2024
Implementation plan and Establish system environment	N	1	1/10/2024	1/10/2024
Implementation	O	10	1/10/2024	10/10/2024
Testing and Maintenance	P	2	12/10/2024	14/10/2024
System Documentation (Q)	Q	1	16/10/2024	16/10/2024
Presentation (R)	R	1	17/10/2024	17/10/2024

TechEdu Gannt Chart



Chapter 3: Analysis

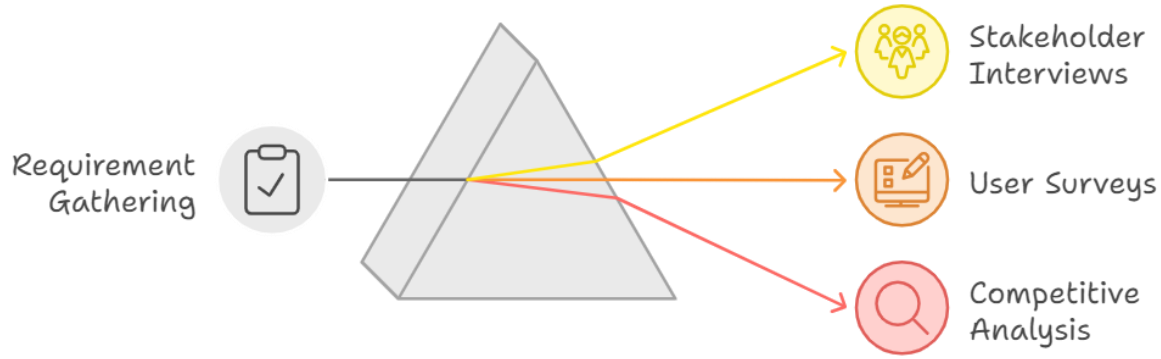
3.1 Requirements Gathering

Identify and collect the requirements needed to build TechEdu. This includes:

Stakeholder Interviews: Conduct interviews with stakeholders to understand their needs and expectations.

User Surveys: Gather input from potential users to identify desired features and functionalities.

Competitive Analysis: Analyze similar platforms to identify gaps and opportunities.



3.2 Functional Requirements

List the core functionalities that the system must have. Examples include:

User Authentication: Registration, login, and password recovery.

Course Management: Creation, modification, and deletion of courses.

User Dashboard: Personalized dashboard for users to track their progress.

Payment Gateway: Integration with payment systems for course enrollment.

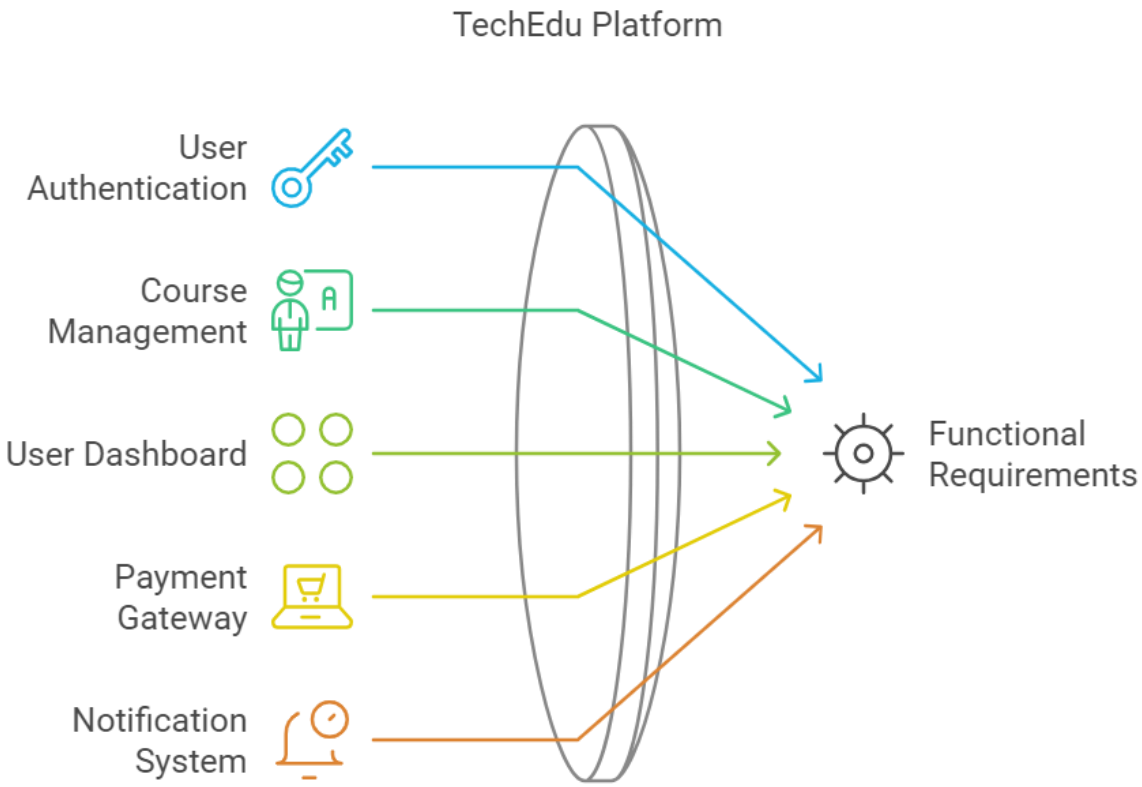
Notification system: Email and in app notifications for updates and reminders.

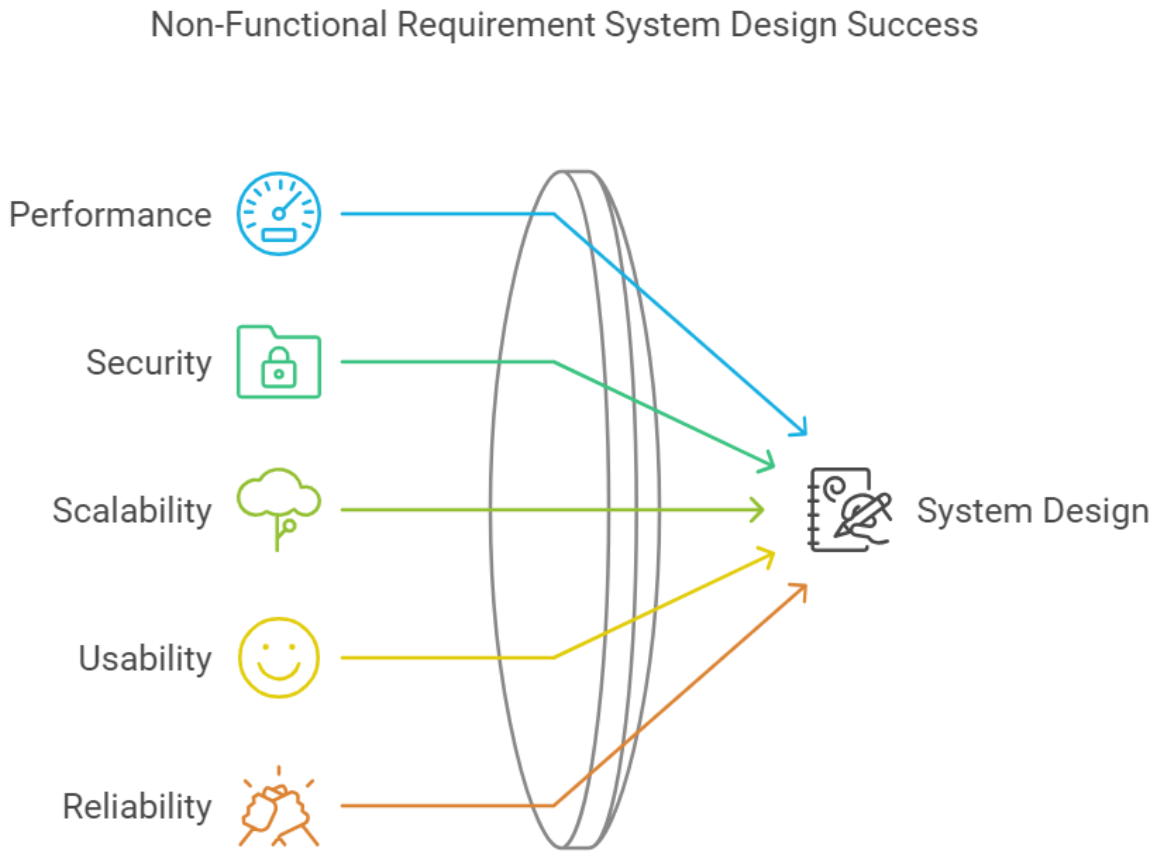
3.3 Non-Functional Requirements

Outline the non-functional requirements that the system must meet. Examples include:

- **Performance: The system should handle a certain number of concurrent users.**

- **Security: User data must be protected with encryption and secure access controls.**
- **Scalability: The system should be able to scale to accommodate growing user base.**
- **Usability: The interface should be user-friendly and accessible.**
- **Reliability: The system should have high uptime and quick recovery from failures.**

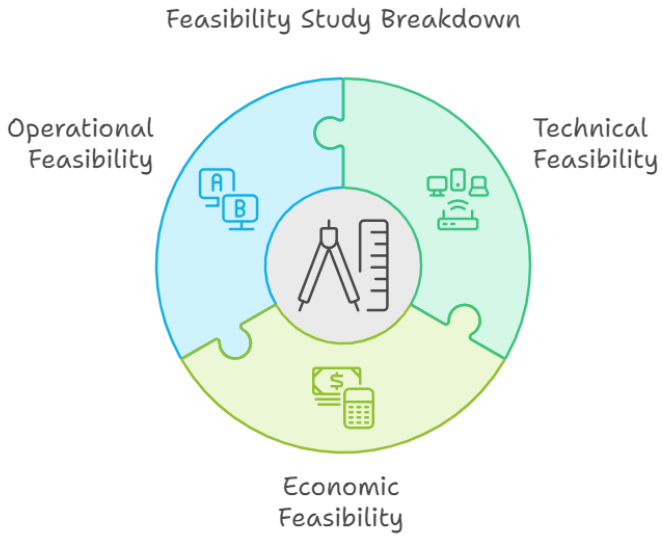




3.4 Feasibility Study

Evaluate the feasibility of the project based on various factors:

- **Technical Feasibility: Assess whether the required technology and expertise are available.**
- **Economic Feasibility: Analyze the cost-benefit ratio to ensure the project is financially viable.**
- **Operational Feasibility: Determine whether the system can be integrated into existing processes and workflows.**



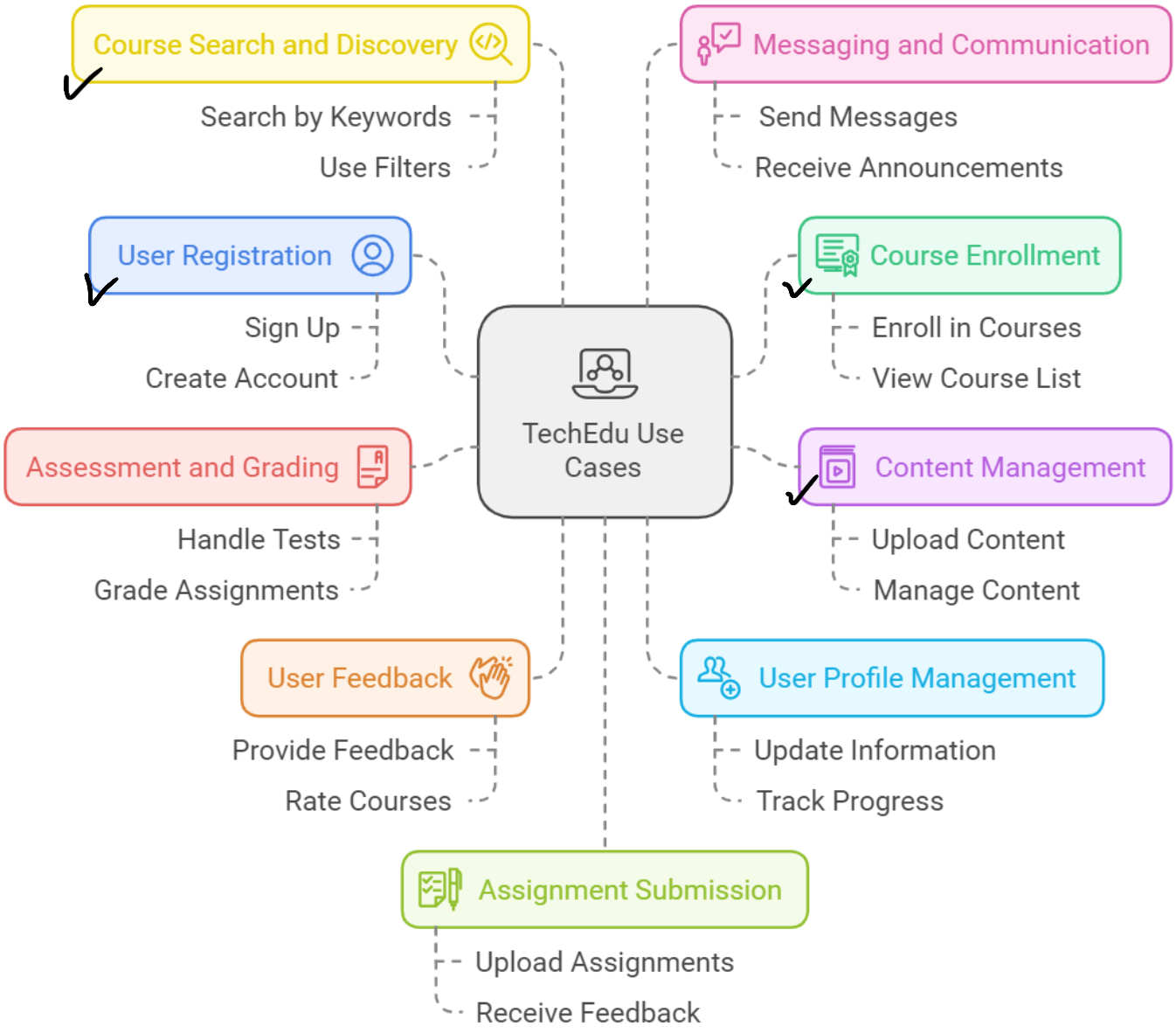
3.5 Use Case Analysis

Define the use cases for TechEdu, detailing how users will interact with the system. Examples include:

- **User Registration: How users sign up and create accounts.**
- **Course Enrollment: The process of enrolling in courses.**
- **Content Management: How educators upload and manage course content.**
- **Assessment and Grading: How tests and assignments are handled.**
- **User Feedback: How users provide feedback and rate courses.**
- **User Profile Management: Users can update their personal information, such as name, email, and profile picture and users can track their learning progress and view course completion certificates.**
- **Course Search and Discovery: Users can search for courses using keywords, categories, or filters (e.g., difficulty level, duration).and Users can view course details, including descriptions, instructor information, and reviews.**
- **Messaging and Communication: Users can send messages to instructors or peers for questions and discussions. And Educators can send announcements and updates to enrolled students.**
- **Assignment Submission: Students can upload assignments and projects for evaluation. And Educators can provide feedback and grades on submitted assignments.**
- **Payment and Subscription Management: Users can purchase individual courses or subscribe to membership plans. Users can view payment history and manage billing information.**
- **Payment and Subscription Management: Users can purchase individual courses or subscribe to membership plans. Users can view payment history and manage billing information.**
- **Progress Tracking and Analytics: Users can view their learning analytics, such as time spent on courses, quiz scores, and completion rates. Educators can track student performance and engagement metrics.**
- **Progress Tracking and Analytics: Users can view their learning analytics, such as time spent on courses, quiz scores, and**

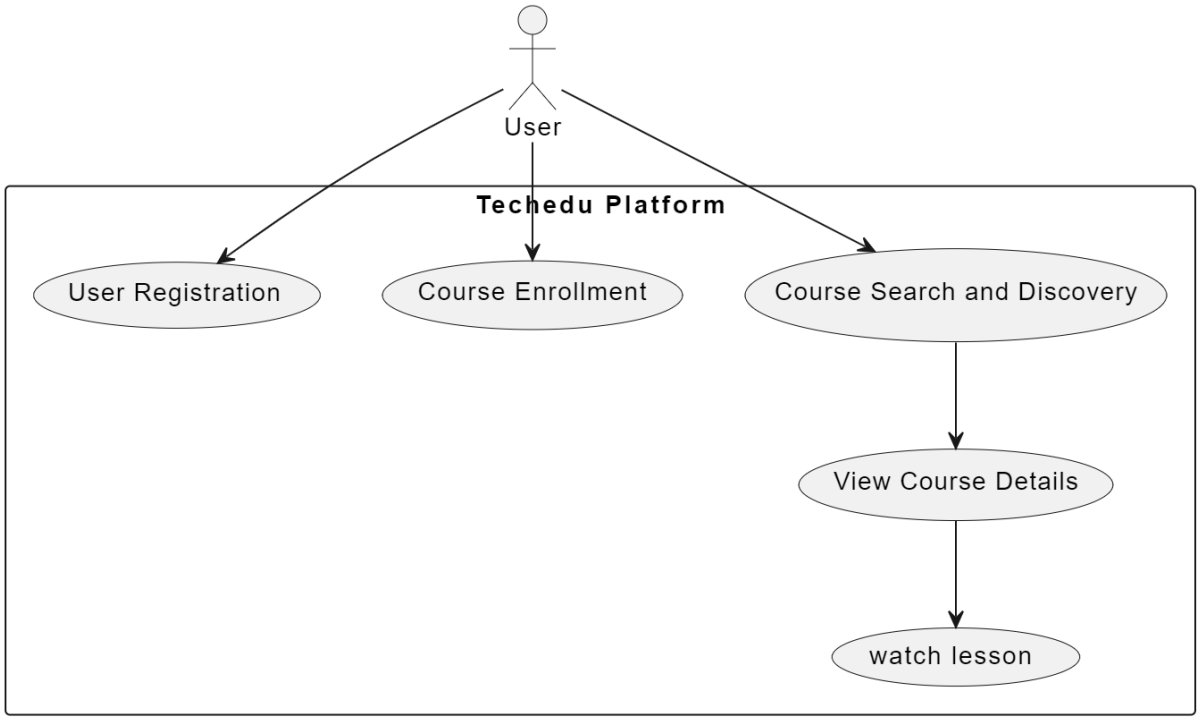
completion rates. Educators can track student performance and engagement metrics.

- **Progress Tracking and Analytics: Users can view their learning analytics, such as time spent on courses, quiz scores, and completion rates. Educators can track student performance and engagement metrics.**

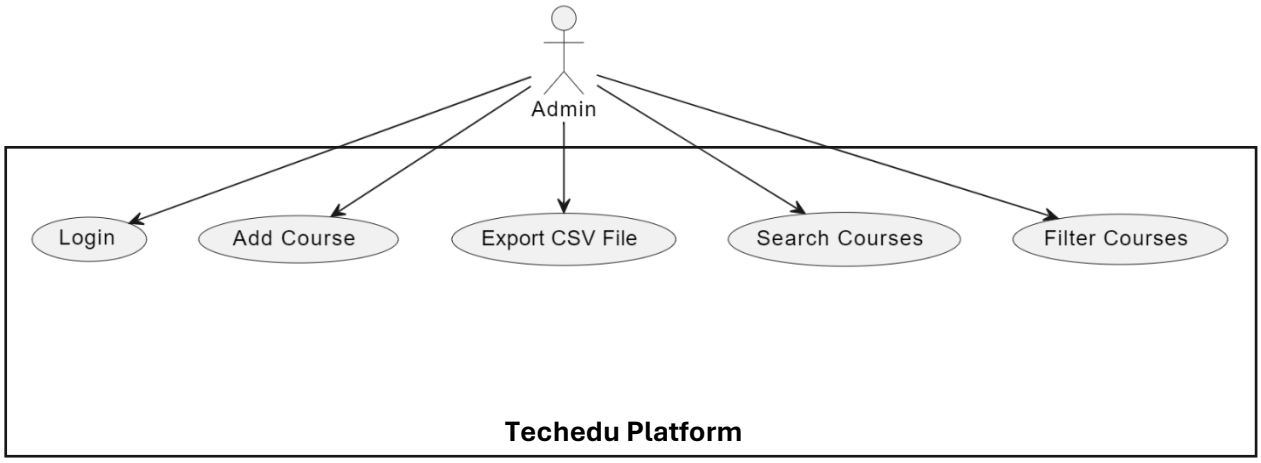


Some use cases used in our project

1.use case for the user of TechEdu

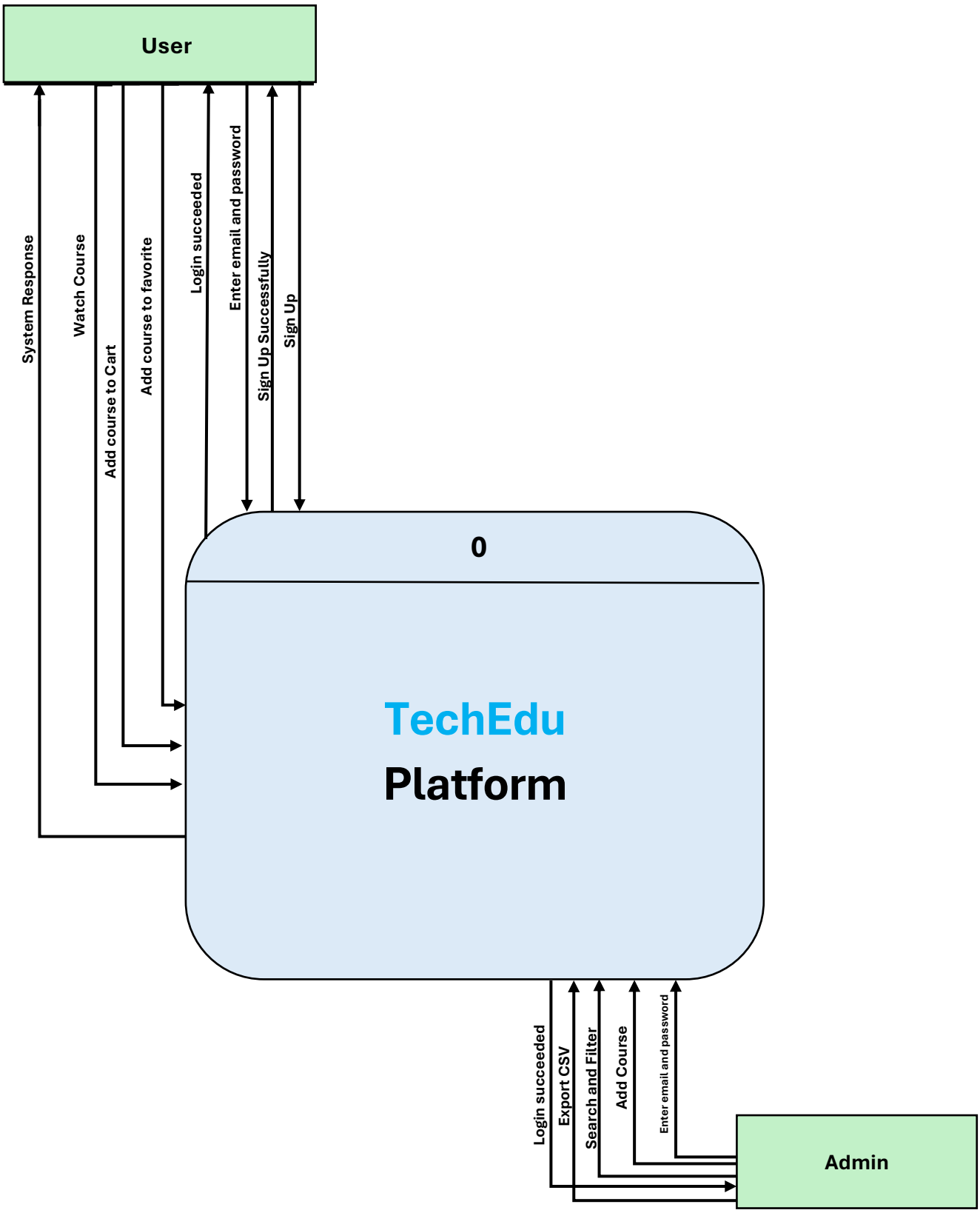


2.use case for the Admin of TechEdu

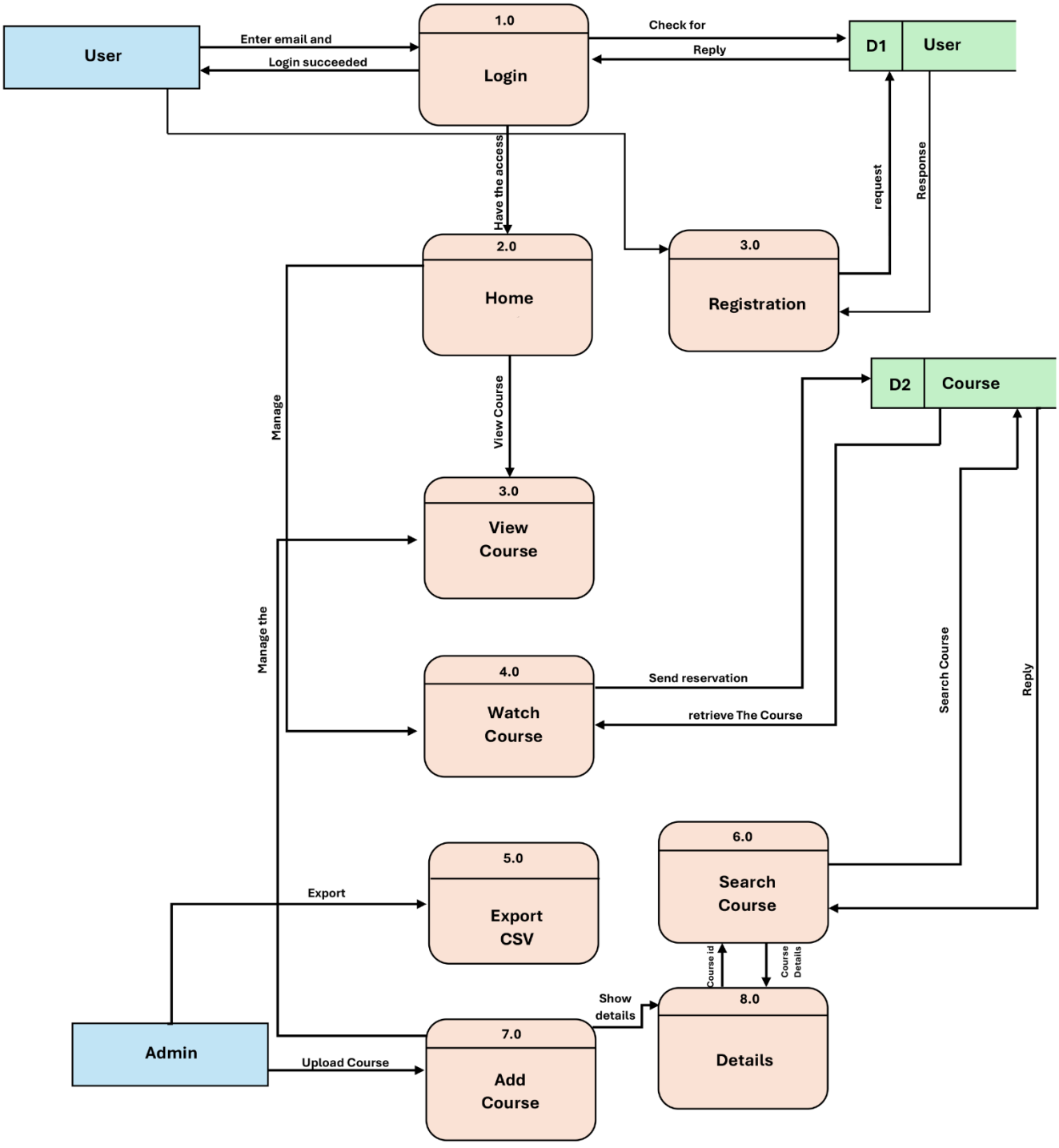


3.6 TechEdu Diagrams

Context Diagram of TechEdu



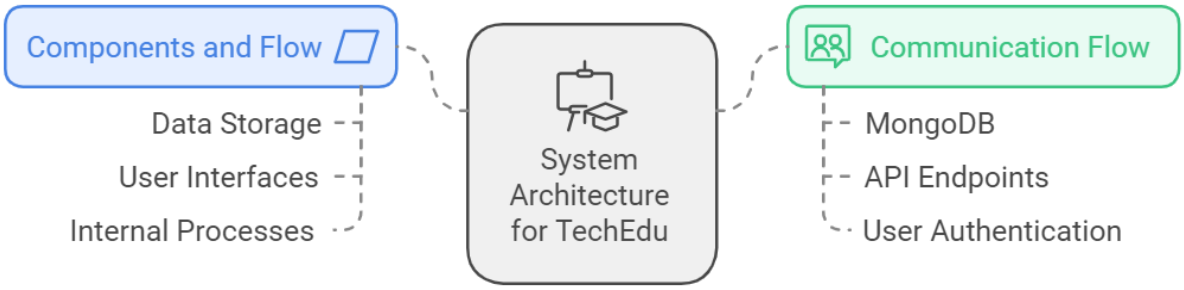
Data Flow Diagram (DFD) for TechEdu



Chapter4: Design

4.1 System Architecture

System Architecture for TechEdu built from 2 things Components and Flow and communication flow



1) Components and Flow

Client Frontend (Vite + React):

- **Role: Handles user interactions and displays educational content.**
- **Responsibilities:**
  - **User registration and authentication**
  - **Displaying available courses**
  - **Enrolling in courses**
  - **Viewing course content**
  - **Interacting with quizzes and assignments**

Backend API (Node.js + Express):

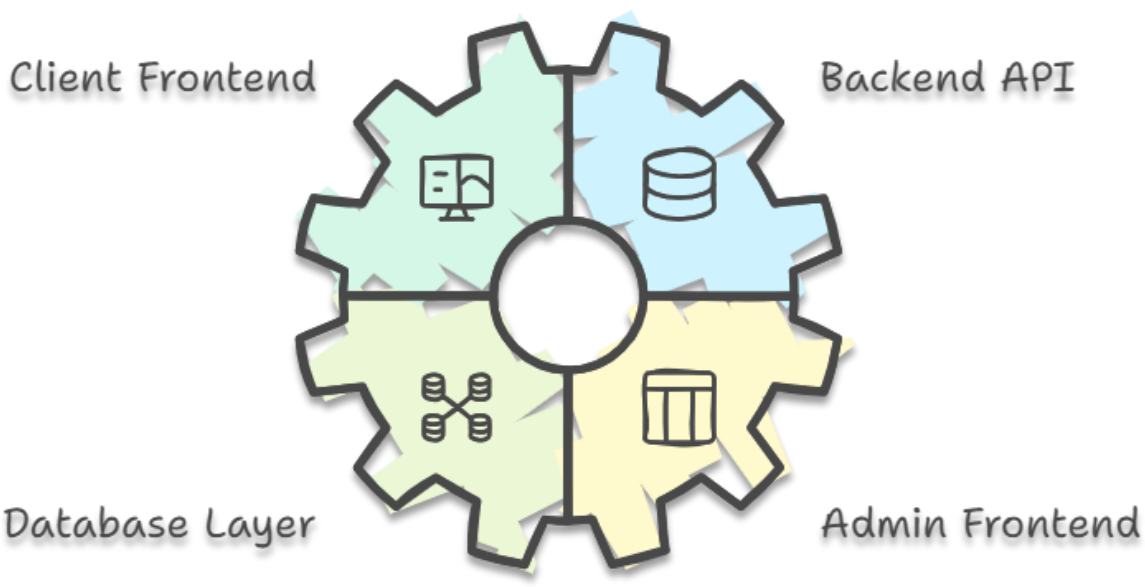
- **Role: Serves as the intermediary between the frontends and the database.**
- **Responsibilities:**
  - **User authentication and authorization**
  - **Handling API requests from client and admin frontends**
  - **Managing courses, users, and other entities**
  - **Communicating with MongoDB for data storage and retrieval**

Database Layer (MongoDB):

- **Role: Stores all persistent data for the application.**
- **Responsibilities:**
  - **Storing user information**
  - **Managing course data**
  - **Keeping track of enrollments, quizzes, assignments, etc.**
  - **Ensuring data consistency and reliability**

Admin Frontend (React.js):

- **Role: Provides an interface for administrators and educators to manage the platform.**
- **Responsibilities:**
  - **Creating and managing courses**
  - **Monitoring user activity**
  - **Managing content, quizzes, and assignments**
  - **Viewing analytics and reports**
  - **Handling user inquiries and support**



2) Communication Flow:

Client Requests

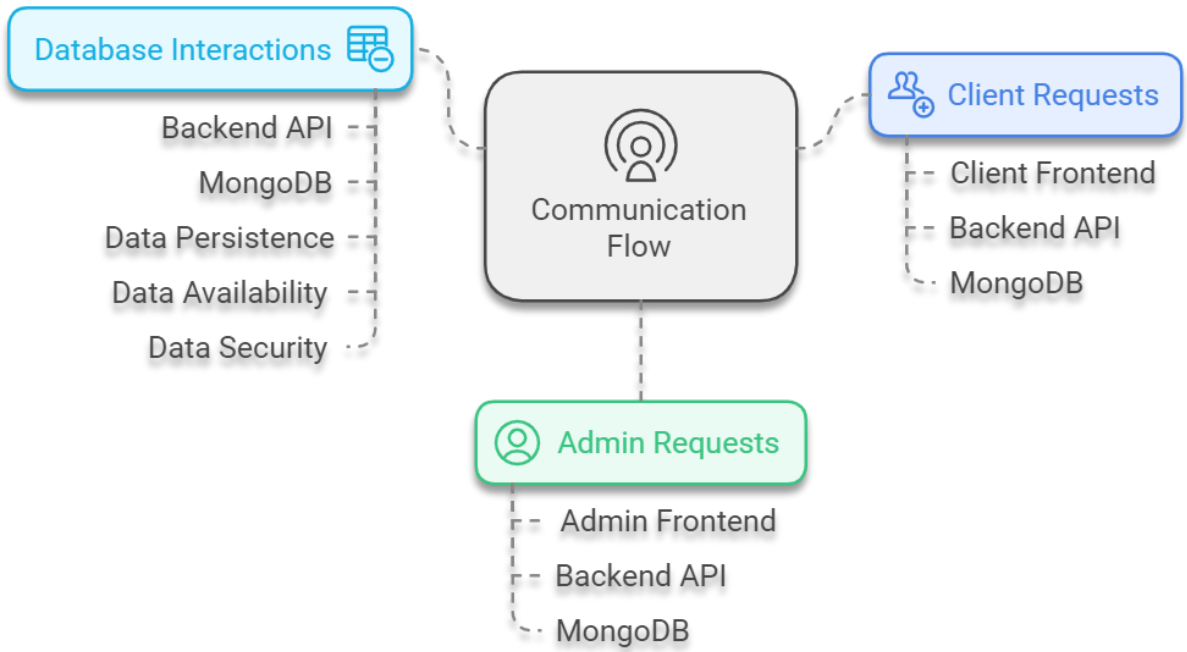
- **Users interact with the Client Frontend, which sends API requests to the Backend API.**
- **The Backend API processes the requests, fetches or updates data in MongoDB, and returns the response to the Client Frontend.**

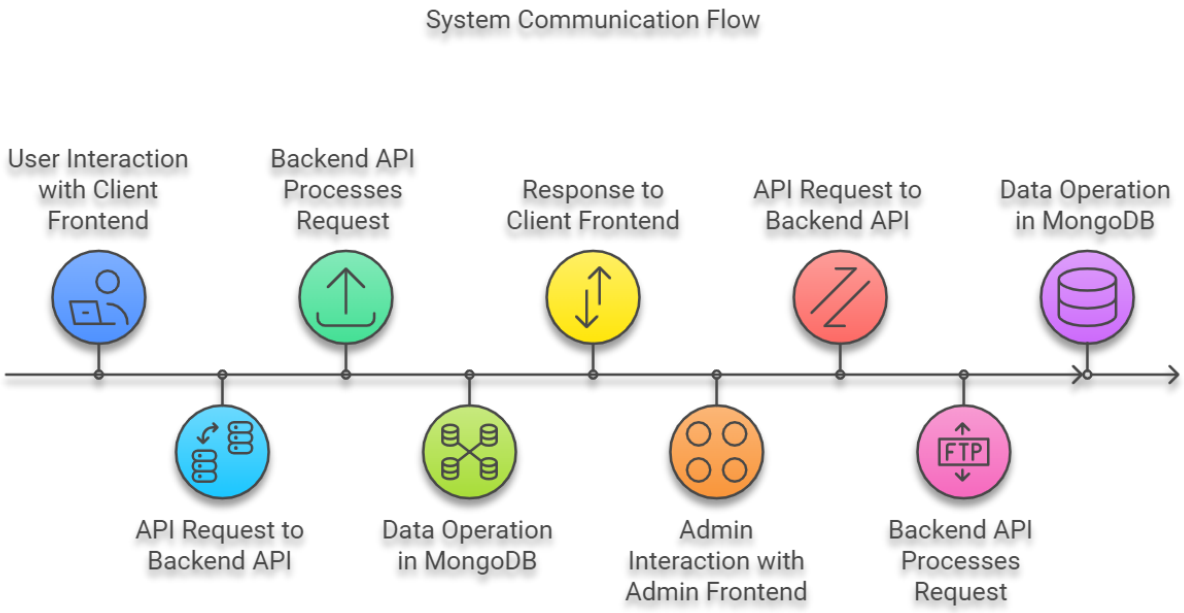
Admin Requests:

- **Administrators and educators interact with the Admin Frontend, sending API requests to the Backend API.**
- **The Backend API handles these requests, performs necessary operations on MongoDB, and returns the response to the Admin Frontend.**

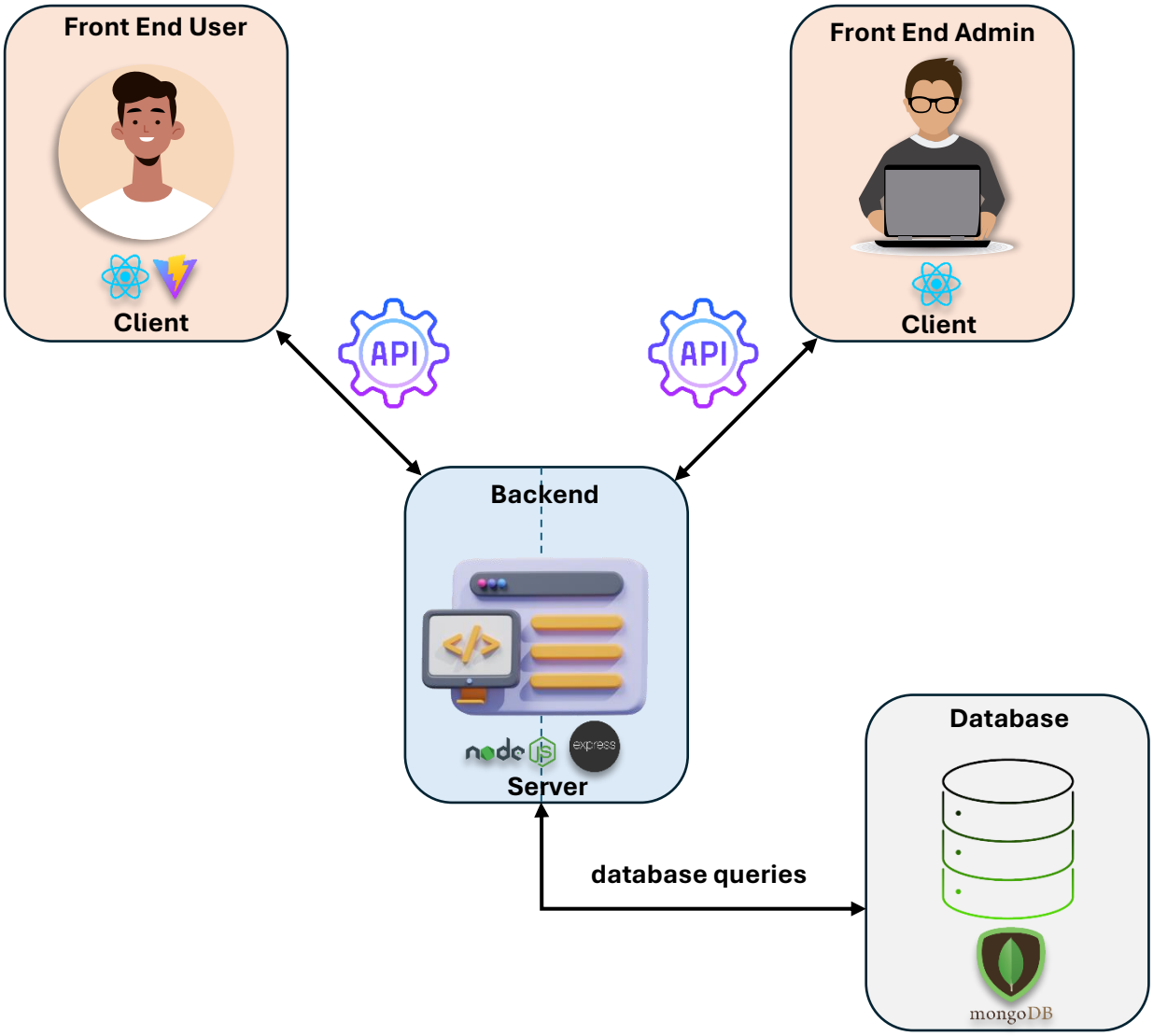
Database Interactions:

- **The Backend API communicates with MongoDB to store and retrieve data as needed.**
- **MongoDB ensures data persistence, availability, and security.**

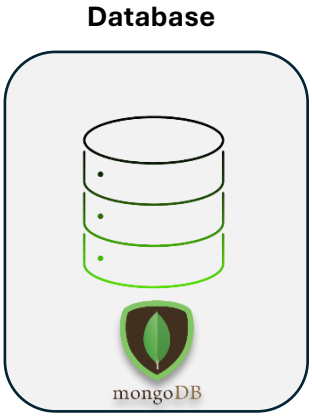




TechEdu Block Diagram



4.2 Database Design



Building Block of database

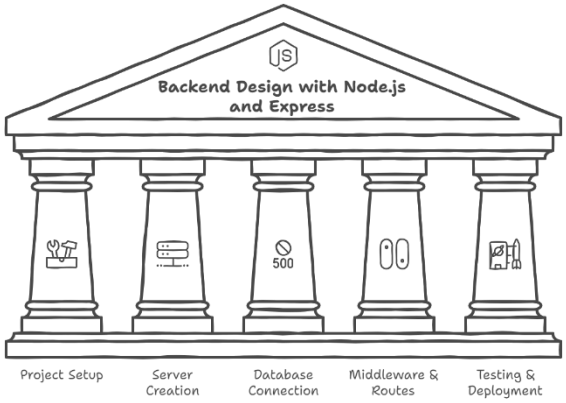
Our document model diagram visually represents the structure and relationships within your database system. This diagram acts as a blueprint, outlining the various entities involved, their attributes, and how they interact with one another. It's an essential tool in the database design process, providing a high-level overview that helps developers understand the data architecture, ensure data integrity, and streamline the implementation process.

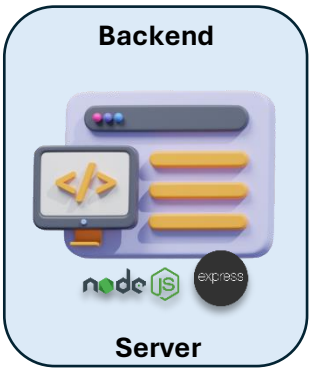
User
id (Primary) fname email mobile password cpassword
tokens (Array) carts (Array) favourites (Array)

Course
id (Primary) cname Iname description content categoryCourse courseType price status courseProfile courseVideo dataCreated dataUpdated

Admin
id (Primary) email password

4.3 Backend Design





Backend design using Node.js and Express is all about creating a robust, scalable, and efficient server-side application. Here's a breakdown:

1) Setting Up Your Project:

- **Initialize Project: Start by creating a new project and initializing it with npm. (`npm init -y`)**
- **Install Dependencies: Add Express and other necessary packages. (`npm install express mongoose dotenv bcryptjs jsonwebtoken`)**

2) Organizing Your Project Structure:

- **Folders: Organize your project with folders like models, routes, controllers, and middleware.**
- **Main File: Create an entry point, typically server.js.**

3) Creating the Server:

- **Set up Express in your server.js.**

4) Database Connection:

- **Mongoose: Use Mongoose to connect to MongoDB and define your schemas.**

5) Middleware:

- **Authentication: Implement middleware for authentication using JWT.**

6) Routes and Controllers:

- **Routes: Define routes in the routes folder, linking them to controller functions.**

- **Controllers: Implement the business logic in the controller's folder.**

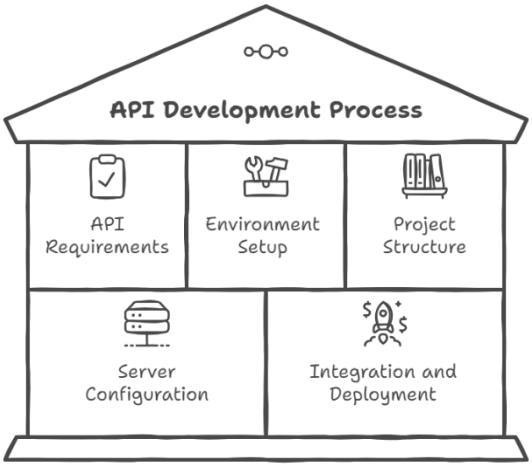
7) Error Handling:

- **Implement global errors by handling middleware to manage errors gracefully.**

8) Testing and Deployment:

- **Use tools like Postman to test your API endpoints.**
- **Write unit tests for your functions.**
- **Deploy our application using services like Heroku or AWS.**

4.4 API Design



The approach to designing your API with React, Node.js, Express, MongoDB, and HTTP:

1) Define the API Requirements:

- **Determine the data and operations the API will need to handle.**
- **Define the user roles and permissions.**

2) Set Up Your Environment:

- **Initialize a Node.js project. (`npm init -y`)**
- **Install necessary packages. (`npm install express mongoose dotenv bcryptjs jsonwebtoken cors`)**

3) Create the Project Structure:

- **Set up folders for models, routes, controllers, middleware, and config.**

- **Create an entry point (server.js).**

4) Configure the Server:

- **set up Express in server.js.**

5) Define Mongoose Schemas:

- **Create schemas for your data models (User, Course, Admin) in the model's folder.**

6) Implement Controllers:

- **Write business logic in the controller's folder.**

7) Set Up Routes:

- **Define API endpoints in the routes folder, linking to controller functions.**

8) Authentication and Authorization:

- **JWT for authentication.**

9) Error Handling:

- **Implement global errors handling middleware.**

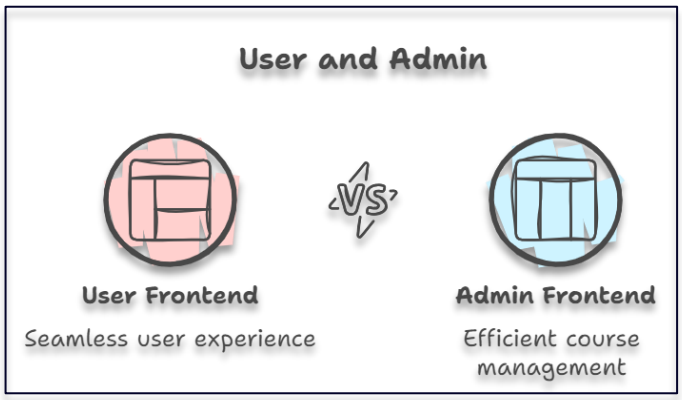
10) Integrate with React:

- **Use Axios or Fetch API in your React app to make HTTP requests to your API.**

Testing and Deployment:

- **Test your endpoints using tools like Postman.**
- **Write unit tests.**
- **Deploy our backend using services like Heroku or AWS.**

4.6 User Interface Design



In the UI design phase, we focus on crafting a seamless and intuitive experience for both the user and admin interfaces. Here's how you can approach it:

User Frontend

1) **Research and Wireframing:**
   - **Understand user needs and create wireframes.**
   - **Focus on key functionalities: browsing courses, managing accounts, and interacting with the platform.**
2) **Design Principles:**
   - **Keep it clean, intuitive, and visually appealing.**
   - **Ensure consistency in design elements like colors, fonts, and buttons.**
3) **Prototyping:**
   - **Use tools like Figma or Sketch to create interactive prototypes.**
   - **Test usability with potential users for feedback.**
4) **Responsive Design:**
   - **Ensure the design is responsive across devices (mobile, tablet, desktop).**
   - **Use a mobile-first approach for accessibility.**

Admin Frontend

1) **Admin Control Courses:**
   - **Upload course , export course as csv to make analysis on the data.**
   - **Focus on key functionalities: browsing courses, managing accounts, and interacting with the platform.**

TechEdu

User Interface & Experience Design

User Before LOGIN

TechEdu

We Own The Future

An Integrated Platform That has Everything a Student Needs To Succeed

Register Now

A New Journey with Skill Courses From Zero To Hero

Explore Our World

Top Watched Courses

Institut Mines-Télécom

5G Network Fundamentals

Course

Qualcomm Academy

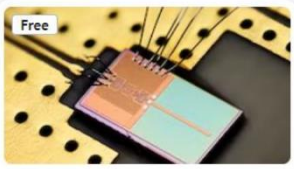
5G for Everyone

Course

Institut Mines-Télécom

4G Network Fundamentals

Course



Eindhoven University of Technology

Microwave engineering and antennas

Course

Yonsei University

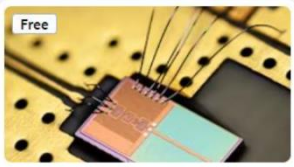
Wireless Communications for Everybody

Course

Edge Impulse

Introduction to Embedded Machine Learning

Course



Eindhoven University of Technology

RF and millimeter-Wave Circuit Design

Course

EDUCBA

Introduction to AutoSAR

Course

Our Vision

We envision a world where anyone, anywhere, has the power to transform their lives through learning.

We believe

Learning is the source of human progress.

300+ world-class partners are teaching the world.

"Coursera has truly helped to democratize education. I was born in Trinidad and Tobago, and I have lived in Jamaica and the United States. I have received emails from students in all three locations who have taken my course on the Coursera platform. I don't believe Coursera is focused on education alone. Coursera is in the business of reaching learners and changing lives."

Hayden Noel,
Clinical Associate Professor of Business Administration, University of Illinois at Urbana-Champaign

Our story

Coursera was founded by Daphne Koller and Andrew Ng in 2012 with a vision of providing life-transforming learning experiences to learners around the world. Today, Coursera is a global platform for online learning and career development that offers anyone, anywhere, access to online courses and degrees from leading universities and companies. Coursera received B Corp certification in February 2021, which means that we have a legal duty not only to our shareholders, but to also make a positive impact on society more broadly, as we continue our efforts to reduce barriers to world-class education for all. 148 million learners and more than 7,000 campuses, businesses, and governments have come to Coursera to access world-class learning—anytime, anywhere.

Contact Us

Have any questions? The quickest way to get in touch with us is using the contact information below.

Privacy Inquiries

If you have questions about our Privacy Notice or an enquiry about how we protect your personal information, you can contact us at privacy@TechEdu.org.

Special Concerns

Security vulnerabilities on the Coursera site may be reported via the HackerOne platform. We take our site security and user privacy very seriously, and we appreciate your help in keeping Coursera safe!

Our Offices

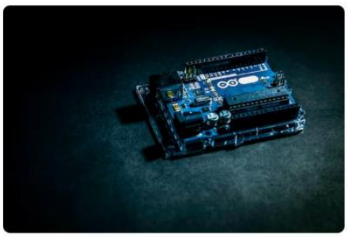
San Francisco, CA

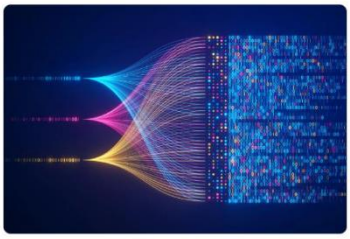
Denver, CO

Dublin, Ireland

Cource Categories

All | AI | System Design | Web Programming | Backend | Design





Front End web Development

Learn HTML & CSS

$300

Front End web Development

JavaScript Language

$600

Front End web Development

Typescript

$200

Front End web Development

Bootstrap

$24

Front End web Development

React

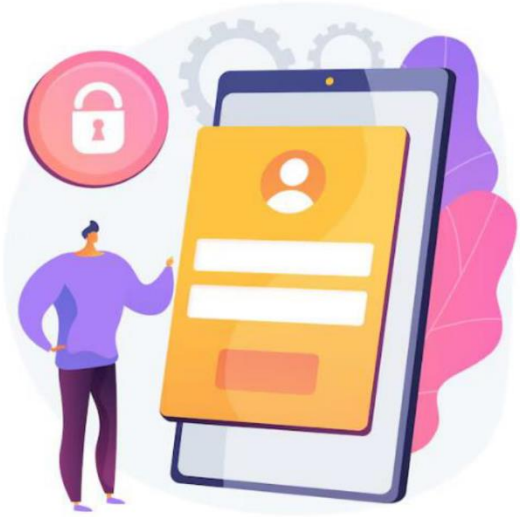
$14

Front End web Development

Angular

$12

Register Now



Username

Email

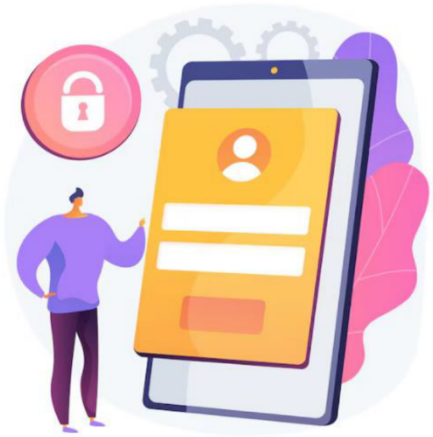
Password

Confirm Password

Register

User After Sign Up

Login Now



Email

Password

Forget Password?

Login

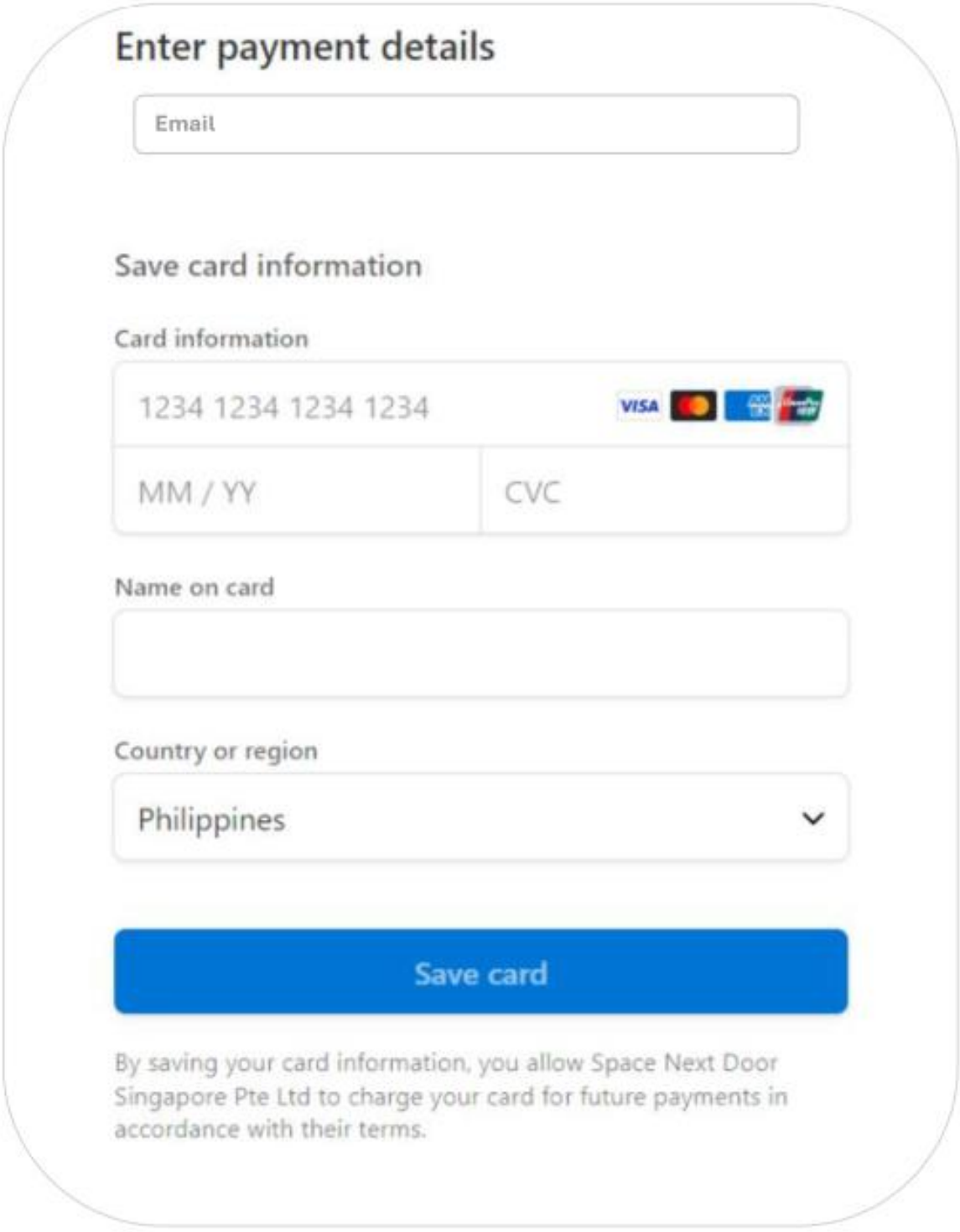
Course	Name	Price	Add to card	Remove
	LEARN HTML & CSS	100$	Add to Card	
	JAVA Script Language	200 $	Add to Card	
	PHP & MYSQL	300$	Add to Card	

Course	Name	Price	Subtotal	Remove
	LEARN HTML & CSS	100$	100$	
	JAVA Script Language	200 $	200 $	
	PHP & MYSQL	300$	300$	

Cart Total

CART SUBTOTAL	**600$**
TOTAL	**600$**

Proceed To Check Out

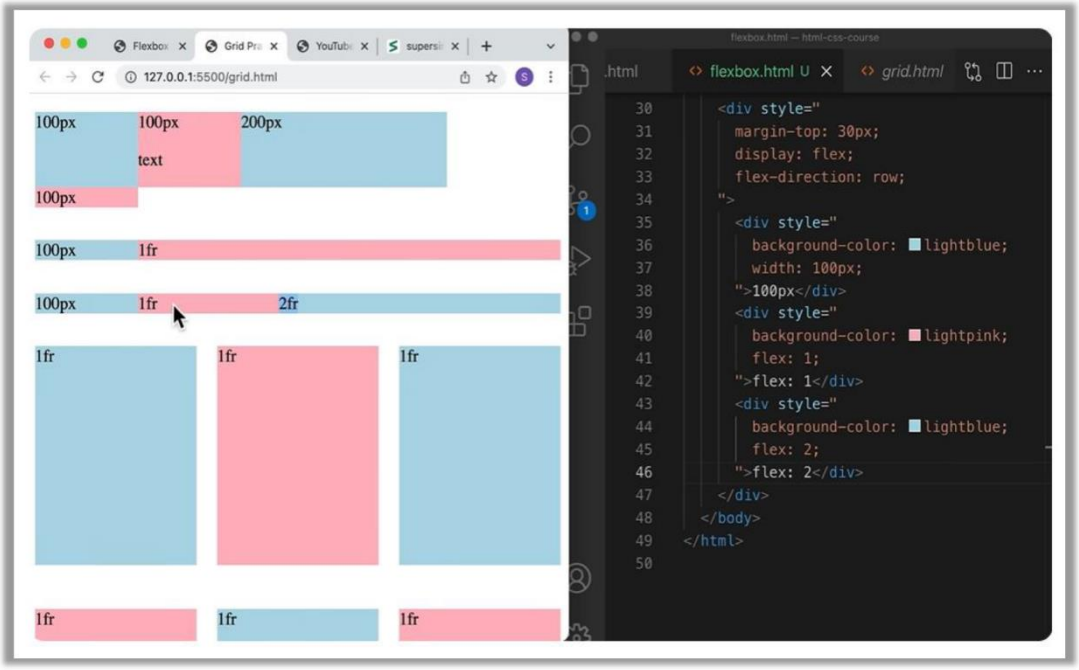


***Hint: After Successfully Payment the Course Will Open**

Course Description

This course is the right start for the web page development field, and this course consists of two parts: The first part provides a comprehensive knowledge of the HTML language, which represents the basics of knowledge of how to develop the web.

View Course



Previous Next/CSS

Admín Panel

Admin Panel

Home

About

Add Course

Welcome Admin

Search....

Search

Export to CSV | Filter By Category Course | Short By Value | Filter By Status

id	Course name	Instructor name	Course Description	Course Content	Type of category Course	Type of Course	Price	Status	Profile Course	Video Course	Action
1	IoT	Ziad	Internet of things	Internet of things	IOT	IOT	800$	Active			No

Admin Panel

Home

About

Add Course

Create Coure

Course Name

Instructor Name

Course Description

Course Content

Select Category

Select Type Of Course

Price

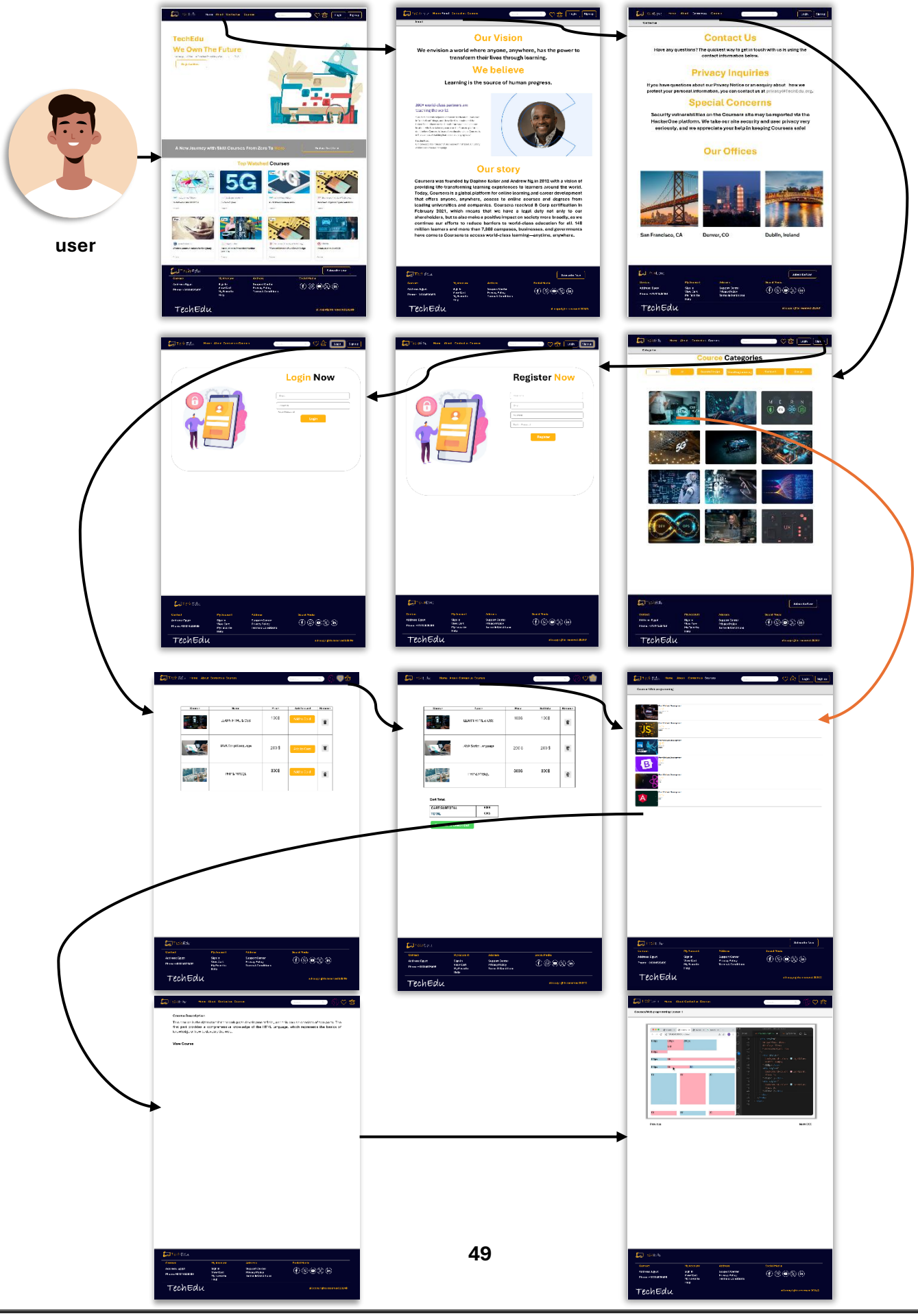
Select Status

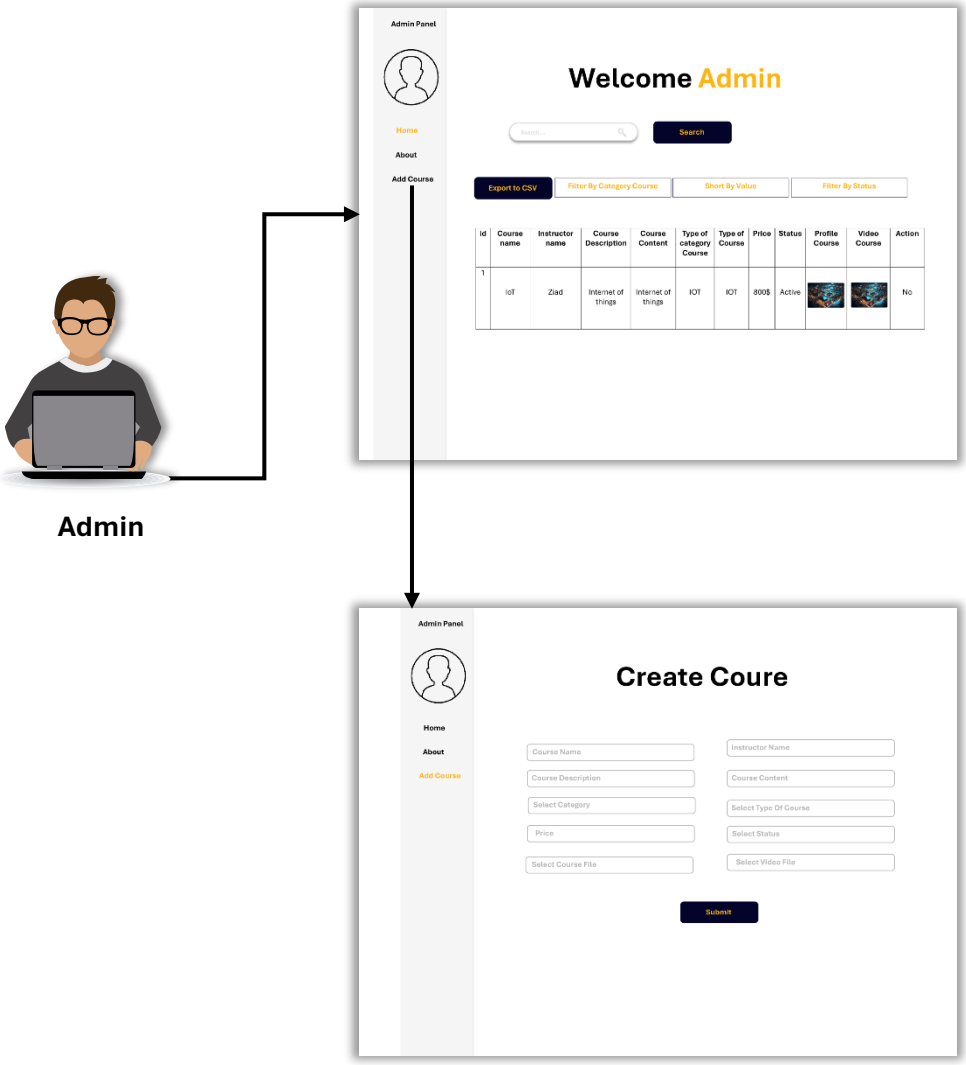
Select Course File

Select Video File

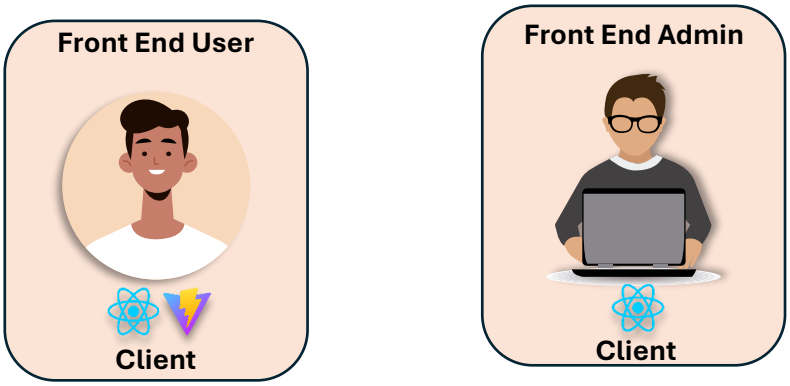
Submit

4.7 Wireframes (System Structure)



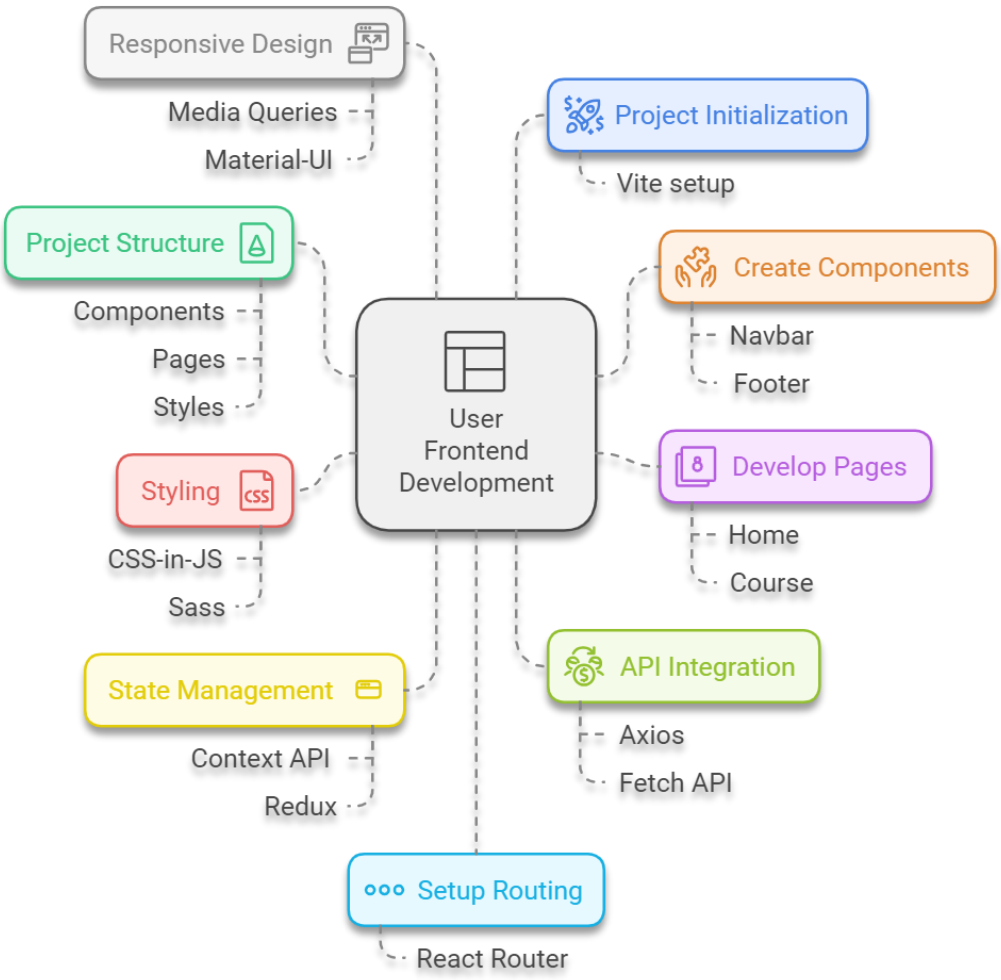

Admin Panel
Home
About
Add Course
Welcome Admin
Search
Export to CSV
Filter By Category Course
Short By Value
Filter By Status
id
Course name
Instructor name
Course Description
Course Content
Type of category Course
Type of Course
Price
Status
Profile Course
Video Course
Action
1
IoT
Ziad
Internet of things
Internet of things
IOT
IOT
800$
Active
No
Admin
Admin Panel
Home
About
Add Course
Create Coure
Course Name
Instructor Name
Course Description
Course Content
Select Category
Select Type Of Course
Price
Select Status
Select Course File
Select Video File
Submit

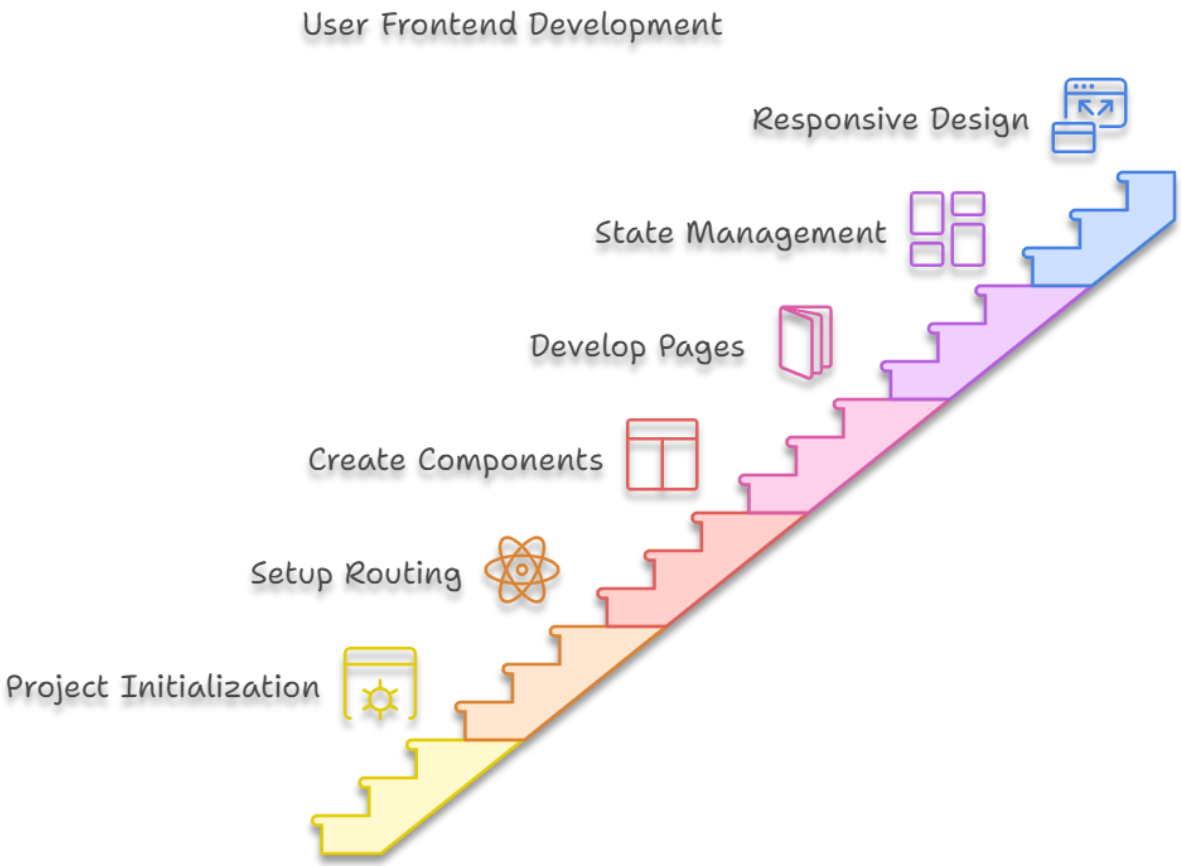
4.8 Frontend Design



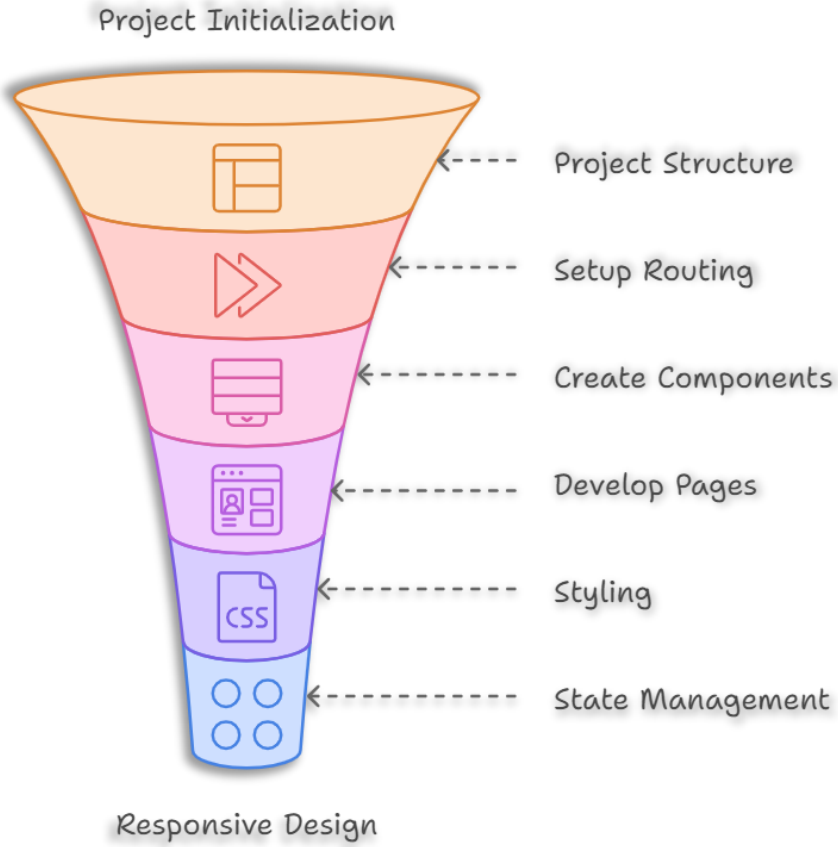
let's map out the design sequence for both the user and admin frontends using React. We'll start with the user frontend using React and Vite, then move on to the admin frontend using React.

User Frontend with React and Vite

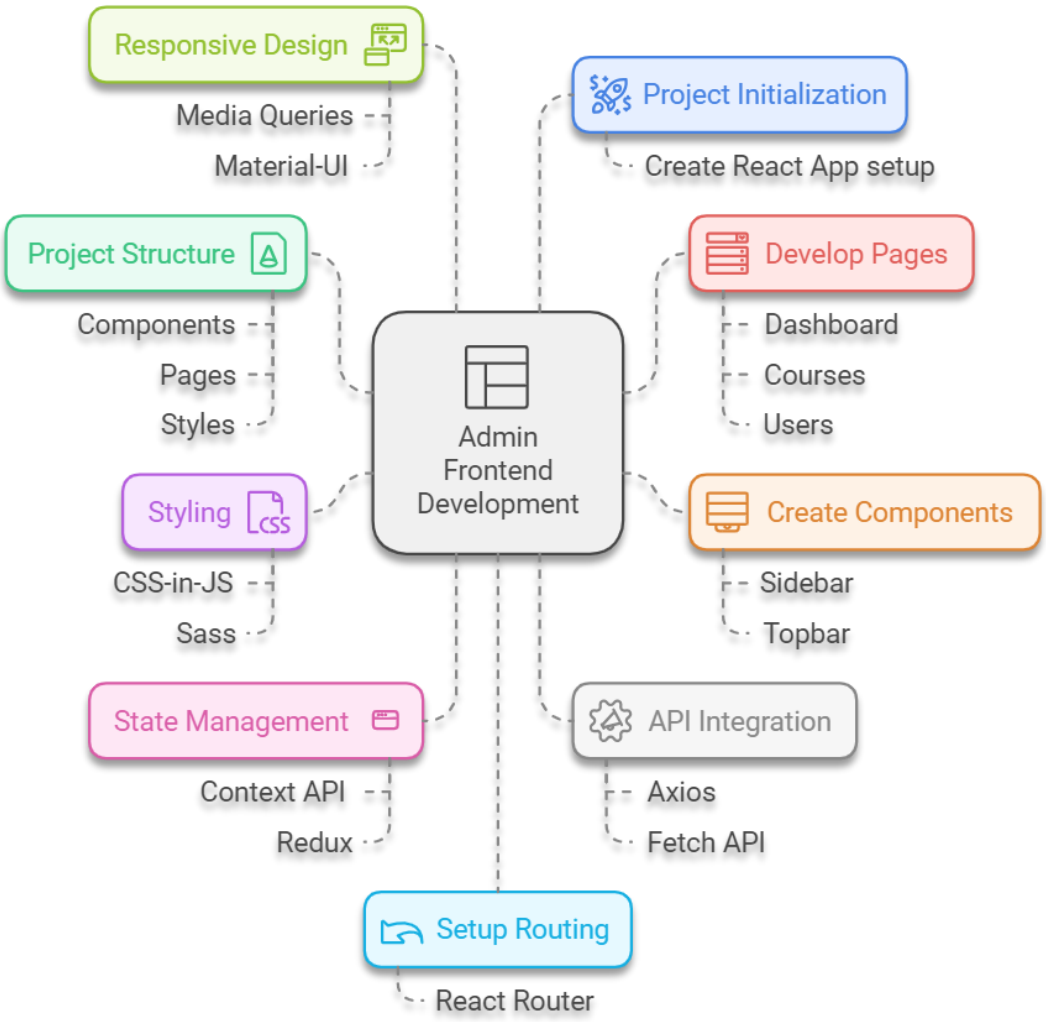


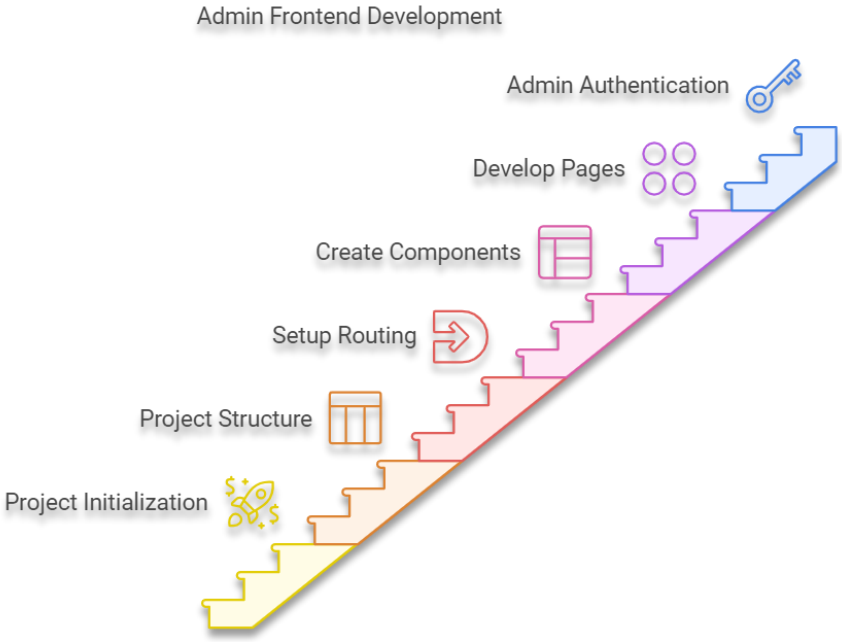


User Frontend Development Funnel



Admin Frontend with React





Chapter 5: Implementation

1. Frontend Development

User Frontend (Vite + React)

1. **Project Initialization:**
    - **Initialize with Vite.**
    - **Install dependencies.**
2. **Project Structure:**
    - **Organize folders: components, pages, styles, services, assets.**
3. **Setup Routing:**
    - **Configure routes with React Router.**
4. **Create Components:**
    - **Reusable components like Navbar, Footer, CourseCard, etc.**
5. **Develop Pages:**
    - **Home, Course, Profile, Cart, etc.**
6. **Styling:**
    - **CSS-in-JS, Sass for polished styling.**
7. **State Management:**
    - **Context API or Redux.**
8. **User Authentication:**
    - **Implement JWT-based authentication.**
9. **API Integration:**
    - **Use Axios or Fetch API.**
10. **Responsive Design:**
    - **Ensure accessibility across devices.**

Admin Frontend (React)

1. **Project Initialization:**

- Initialize with Create React App.
- Install dependencies.

2. Project Structure:
   - Organize folders: components, pages, styles, services, assets.
3. Setup Routing:
   - Configure routes with React Router.
4. Create Components:
   - Sidebar, Topbar, CourseTable, UserTable, etc.
5. Develop Pages:
   - Dashboard, Courses, Users, Settings.
6. Styling:
   - CSS-in-JS, Sass for polished styling.
7. State Management:
   - Context API or Redux.
8. Admin Authentication:
   - Implement JWT-based authentication.
9. API Integration:
   - Use Axios or Fetch API.
10. Responsive Design:
    - Ensure accessibility across devices.

2. Backend Development (Node.js + Express)

- Set up server, routing, and middleware.
- Implement business logic in controllers.
- Connect to MongoDB.

3. Database Integration (MongoDB)

- Define Mongoose schemas and models.

- **Implement CRUD operations.**
- **Ensure data validation.**

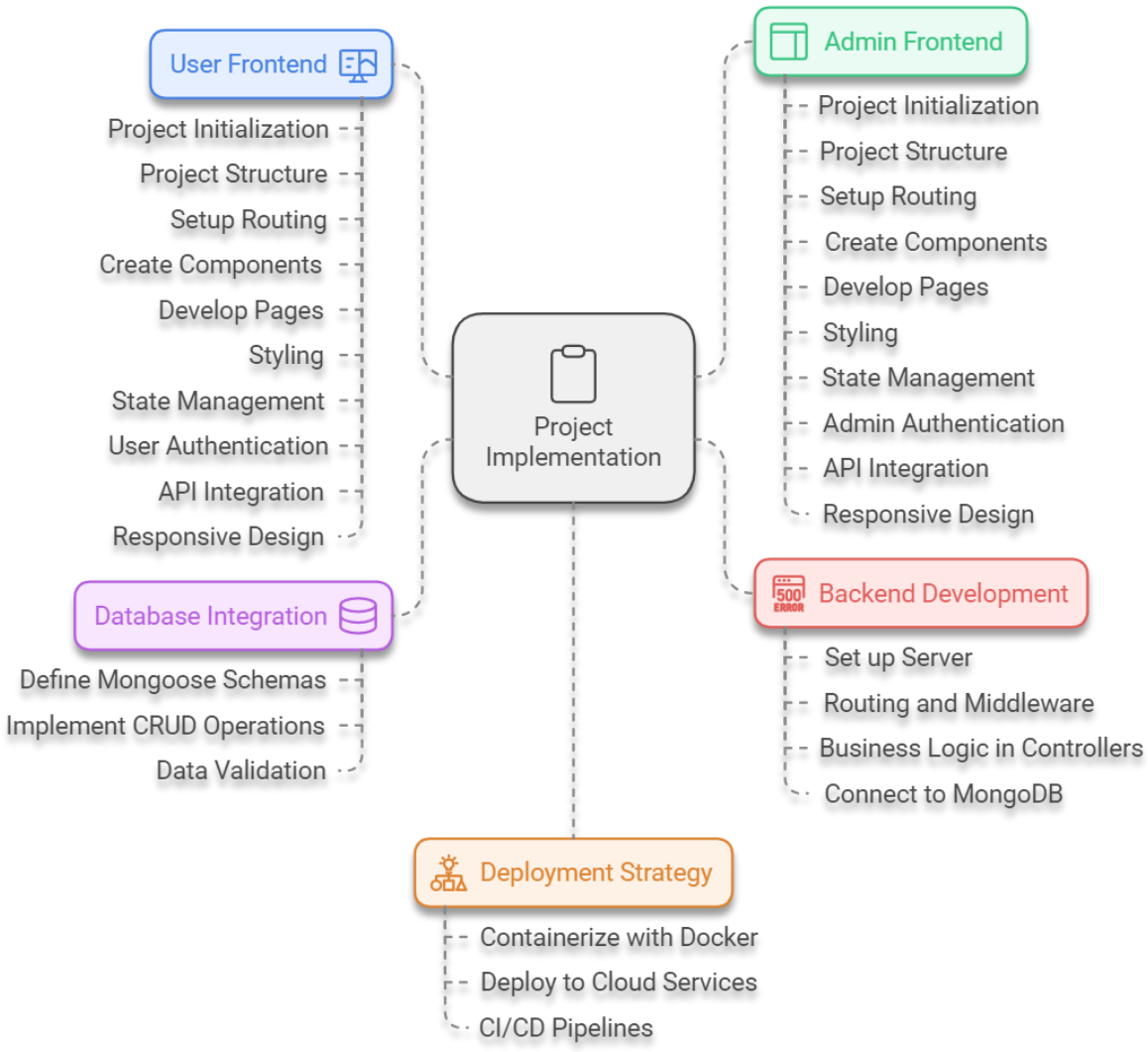
4. Deployment Strategy

- **Containerize with Docker.**
- **Deploy cloud services like AWS and Heroku.**
- **Implement CI/CD pipelines.**

5. Wireframes

- **Detailed wireframes for both user and admin interfaces.**
- **Usability validation with stakeholders.**

This sets a clear path for implementing both user and admin frontends, along with the backend, database integration, and deployment.



Chapter 9: Conclusion