**VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY**

**INTERNATIONAL UNIVERSITY**

 **SCHOOL OF COMPUTER SCIENCE & ENGINEERING**

**WEB APPLICATION DEVELOPMENT**

**IT093IU**

PROJECT REPORT

**CYBERSECURITY ONLINE LEARNING PLATFORM**

**Submitted by**

Nguyễn Quốc Tuấn - ITITIU22177

Date Submitted: 21/10/2023

Date Performed: 3/5/2025

Course Instructor: Dr. Nguyen Van Sinh

**Table of Contents**

[List of Figures iii](#_Toc119612917)

[List of Tables iii](#_Toc119612918)

[I. Introduction 1](#_Toc119612919)

[1. 1st title 1](#_Toc119612920)

[2. 2nd title 1](#_Toc119612921)

[II. Experimental procedure 1](#_Toc119612922)

[1. Materials and equipment 1](#_Toc119612923)

[a. Materials 1](#_Toc119612924)

[b. Equipment 1](#_Toc119612925)

[2. Experimental procedures 2](#_Toc119612926)

[a. Experiment 01’s title 2](#_Toc119612927)

[b. Experiment 02’s title 2](#_Toc119612928)

[c. Experiment 03’s title 2](#_Toc119612929)

[III. Results and discussion 3](#_Toc119612930)

[1. Results 3](#_Toc119612931)

[2. Discussion 3](#_Toc119612932)

[IV. Assignment 4](#_Toc119612933)

[V. Conclusion 4](#_Toc119612934)

[References 5](#_Toc119612935)

**List of Figures**

***(Please note the page number of figures (pictures) that you use in your report. Titles for figures should appear directly below the figures. Remember to delete this note before printing.)***

[**Figure 1.** Erlenmeyer flask 1](#_Toc113474817)

**List of Tables**

***(Please note the page number of tables that you use in your report. Titles for tables should appear directly above the table. Remember to delete this note before printing.)***

[**Table 1.** Experiment 01’s results 3](#_Toc113474722)

# Introduction

* This report presents the design, development, and features of TS Academy, a cutting-edge online platform focused on delivering comprehensive cybersecurity education. As digital threats continue to evolve rapidly, the need for accessible, practical cybersecurity training has never been greater. TS Academy meets this demand by offering an engaging learning environment that blends foundational theory with hands-on practice.
* Catering to learners of all skill levels, from novices exploring digital security fundamentals to experienced users sharpening advanced techniques, TS Academy employs a dual learning approach. Its robust theoretical courses provide a deep understanding of cybersecurity principles, concepts, and best practices, while interactive labs enable learners to apply their knowledge in realistic, simulated environments.

# Project Overview

TS Academy is built on three core pillars:

## Theory Lessons:

The platform delivers structured, engaging lessons on essential cybersecurity concepts, principles, and best practices. These lessons provide learners with a strong knowledge foundation to comprehend digital threats and defense mechanisms, designed to be accessible and relevant for users at any stage of their learning journey.

## Interactive Labs:

To complement theoretical learning, interactive labs offer a hands-on environment where users can put their knowledge into practice. These labs simulate real-world cybersecurity scenarios, allowing learners to test defensive strategies, detect vulnerabilities, and build practical skills in a secure, controlled setting, reinforcing their theoretical understanding.

## Admin Dashboard:

A powerful admin dashboard equips platform administrators with tools to manage operations efficiently. Features include user management, content creation and organization for lessons and labs, progress tracking, and potential report generation, ensuring seamless maintenance and oversight of the learning ecosystem.

# Requirement analysis

## Functional Requirement

### User Management

* + - User registration and authentication system
    - Role-based access control (student,administrator)
    - User profile management with customization options

### Course Management

* + - Course creation and publishing
    - Content organization by categories, subjects, and difficulty levels

### Learning Expierence

* + - Interactive lesson modules with progress tracking

### Interactive Labs

* + - Hands-on virtual lab environments for practical exercises
    - Sandbox environments for experimentation
    - Lab exercise tracking and progress monitoring
    - Simulated environments for real-world scenario practice
    - Lab result submission and feedback system

### Administration Dashboard

* + - User activity monitoring and engagement metrics
    - Content management and moderation tools

## Non-Functional Requirements

### Performance

* + Page load times under 3 seconds
  + Efficient database queries to handle large datasets
  + Scalable architecture to accommodate growth

### Security

* + End-to-end encryption for sensitive data

### Usability

* + Intuitive, user-friendly interface design
  + Consistent navigation and interface patterns
  + Clear error messages

### Compatibility

* + Support for major browsers (Chrome, Firefox, Safari, Edge)

## Technical Requirements

### Development Technologies

* + Front-end: Modern JavaScript framework (React)
  + Back-end: Scalable server-side technology (Node.js)
  + Database: Relational (MySQL)
  + Version control system for collaborative development(Github)

### Potential Future Features

* + AI-powered learning path recommendations
  + Augmented reality/virtual reality learning experiences
  + Advanced analytics with predictive capabilities
  + Blockchain-based credential verification
  + Marketplace for educational resources
  + Peer-to-peer tutoring network
  + Content localization and internationalization

# Design

## Backend Architecture Design

The backend architecture of our online learning platform follows a modular, service-oriented approach built on Node.js. This design ensures scalability, maintainability, and separation of concerns through well-defined components.

### System Architecture

The backend implements a classic MVC (Model-View-Controller) pattern with additional middleware components to handle cross-cutting concerns. The system is organized into logical modules that encapsulate specific functionality domains:

* **Models**: Data schemas and business logic
* **Controllers**: Request handling and response formatting
* **Routes**: API endpoint definitions and request routing
* **Middleware**: Cross-cutting concerns like authentication
* **Config**: Environment and application configuration
* **Utils**: Shared utility functions
* **Content**: Security-focused content resources

### Key Components Design

#### Data Models

The platform's data layer is structured around these primary entities:

#### User Model

* + Manages user profiles, authentication credentials, and role-based permissions
  + Supports multiple user types (student, administrator)
  + Handles user-specific settings and preferences

#### Lesson Model

* + Represents structured learning content organized into courses
  + Supports versioning and content management
  + Includes metadata for categorization and discovery

#### LessonPage Model

* + Represents individual content pages within lessons
  + Supports multiple content formats (text, interactive elements)
  + Tracks content revisions and publishing status

#### UserLessonProgress Model

* + Tracks individual user progress through course materials
  + Records completion status

#### Lab Model

* + Defines interactive environments for hands-on practice
  + Specifies resource requirements and configuration parameters
  + Supports different scenario types and difficulty levels

### Controller Layer

Controllers implement the application's business logic and mediate between the data models and API endpoints:

#### Authentication Controller

* + Manages user registration, login, and session management
  + Implements security measures including password hashing and token validation

#### User Controller

* + Manages user profile operations and settings
  + Handles role-based permission checks
  + Processes user activity and progress data
  + Handles credential updates

#### Lesson Controller

* + Delivers educational content based on user progress
  + Manages content sequencing and prerequisite enforcement

#### Lab Controller

* + Manages the lifecycle of interactive lab environments
  + Processes user actions within lab scenarios
  + Evaluates lab exercise completion and provides guidance

#### Docker Controller

* + Handles container provisioning and management for isolated lab environments
  + Manages resource allocation and cleanup
  + Ensures security boundaries between user environments

### Middleware Components

The middleware layer handles cross-cutting concerns:

#### Authentication Middleware

* + Validates user session tokens
  + Implements role-based access control for protected resources
  + Maintains security context throughout request processing

#### Utility Services

Shared utilities provide common functionality:

#### JWT Utilities

* + Generates and validates JSON Web Tokens for authentication
  + Implements token refresh and expiration logic
  + Provides security context for authorized operations

#### File Utilities

* + Handles secure file operations
  + Manages content uploads and storage
  + Implements access control for protected resources

### API Design

Our online learning platform implements a comprehensive RESTful API architecture organized by functional domains. The API is designed with security, performance, and maintainability as key priorities.

#### API Structure and Implementation

The API is built using Express.js with the following core components:

##### Authentication API

The authentication API handles user identity verification and session management:

router.post("/login", authController.login);

router.post("/register", authController.register);

router.get("/check-auth", checkAuth, authController.getAuthStatus);

router.post("/logout", authController.logout);

These endpoints manage:

* User registration with credential validation
* Secure authentication with JWT token issuance
* Session validation for protected resources
* Secure logout and session termination

##### User Management API

The user management API provides access to user profiles and account settings:

router.get("/home", (req, res) => {

  res.json({ message: "valid", user\_name: req.user\_name });

});

router.get("/account", userController.getUser);

router.post("/update-account", userController.updateUser);

These endpoints handle:

* User profile retrieval
* Account settings updates
* User preference management

##### Learning Content API

The learning content API manages educational resources and user progress:

router.get("/", lessonController.getAllLessons);

router.get("/:id", lessonController.getLessonById);

router.post("/create", lessonController.createLesson);

router.put("/:id", lessonController.updateLesson);

router.delete("/:id", lessonController.deleteLesson);

router.get("/:lessonId/pages/:pageNumber", lessonController.getLessonPage);

router.get("/:lessonId/pages", lessonController.getAllLessonPages);

router.post("/:lessonId/pages", lessonController.createLessonPage);

router.put("/pages/:pageId", lessonController.updateLessonPage);

router.delete("/pages/:pageId", lessonController.deleteLessonPage);

router.post("/progress/:lessonId", lessonController.updateUserProgress);

router.get("/user/progress", lessonController.getUserLessonsWithProgress);

These endpoints provide:

* Comprehensive lesson CRUD operations
* Granular access to lesson content pages
* User progress tracking and management
* Content organization and sequencing

##### Interactive Lab API

The lab API handles the creation and management of hands-on learning environments:

router.get("/", labController.labsWithStatus);

router.get("/verify-flag", labController.verifyFlag);

router.post(

  "/create-container",

  labController.getLabName,

  createContainerController

);

router.post("/getlabname", labController.getLabName);

router.post("/delete-container", deleteContainer);

These endpoints manage:

* Lab environment discovery and status checking
* Container-based lab environment provisioning
* Challenge verification and completion tracking
* Resource cleanup and management

##### API Security Features

The API incorporates several security measures:

###### **Authentication Middleware**:

All protected routes use the checkAuth middleware to validate user sessions:

app.use("/api/route", checkAuth, indexRoutes);

app.use("/api/labs", checkAuth, labRoutes);

app.use("/api/lessons", checkAuth, lessonRoutes);

###### **CORS Protection**:

Strict cross-origin resource sharing policies:

app.use(

  cors({

    origin: "http://localhost:4000",

    methods: ["GET", "POST", "PUT", "DELETE"],

    credentials: true,

  })

);

###### **Error Handling**:

Consistent error responses for invalid routes:

app.use((req, res) => {

  res.status(404).json({ message: "Route not found" });

});

## Frontend Design

The frontend of our online learning platform is built as a single-page application (SPA) using React.js, providing a responsive and dynamic user experience. This is a well-structured React application built with TypeScript and Vite, organized with a feature-based approach.

### Component Design

The frontend implements a component-based architecture with clear separation of concerns:

#### Core Components

* **App Component**: Serves as the application root, managing global state and routing
* **Navigation Component**: Provides consistent navigation across the platform
* **Authentication Components**: Handle user login, registration, and session management
* **Protected Route Component**: Ensures authenticated access to restricted areas

#### Feature Components

##### Lesson Components

* **LessonView**: Displays lesson content to users with navigation controls
* **LessonEditor**: Provides content creation and editing interface for admin
* **LessonCreate**: Facilitates the creation of new educational content
* **LessonCards**: Displays available lessons with progress indicators

##### Lab Components

* **Lab**: Container component for interactive lab environments
* **LabCards**: Selection interface for available lab exercises

##### Admin Interface

* Administrative dashboard for platform management
* User, lesson, and lab management controls

### State Management

The application uses a combination of:

* Local component state with React's useState hook
* Context API for global state management (authentication, user preferences)
* Prop drilling for component communication within feature modules

### Routing Structure

The frontend implements client-side routing with React Router:

* Public routes (Home, About, Login)
* Protected routes requiring authentication (Account, Lessons, Labs, Admin)
* Nested routes for complex features (lesson pages, lab exercises)
* 404 handling for invalid routes

### API Integration

The frontend communicates with the backend through:

* RESTful API calls to the server endpoints
* Authentication token management
* Error handling and loading states

### Performance Considerations

Several strategies are employed to optimize performance:

* Code splitting for route-based lazy loading
* Asset optimization for efficient delivery
* Memoization for expensive computations
* Efficient rendering of lists and dynamic content

### Service Layer Architecture

The application implements a robust service layer that acts as an intermediary between the UI components and backend APIs:

#### Core Services

* **authService**: Manages authentication flows, token handling, and user session persistence
* **accountService**: Handles user profile operations, preferences, and account management
* **adminService**: Provides administrative functionality and system configuration operations

#### Feature Services

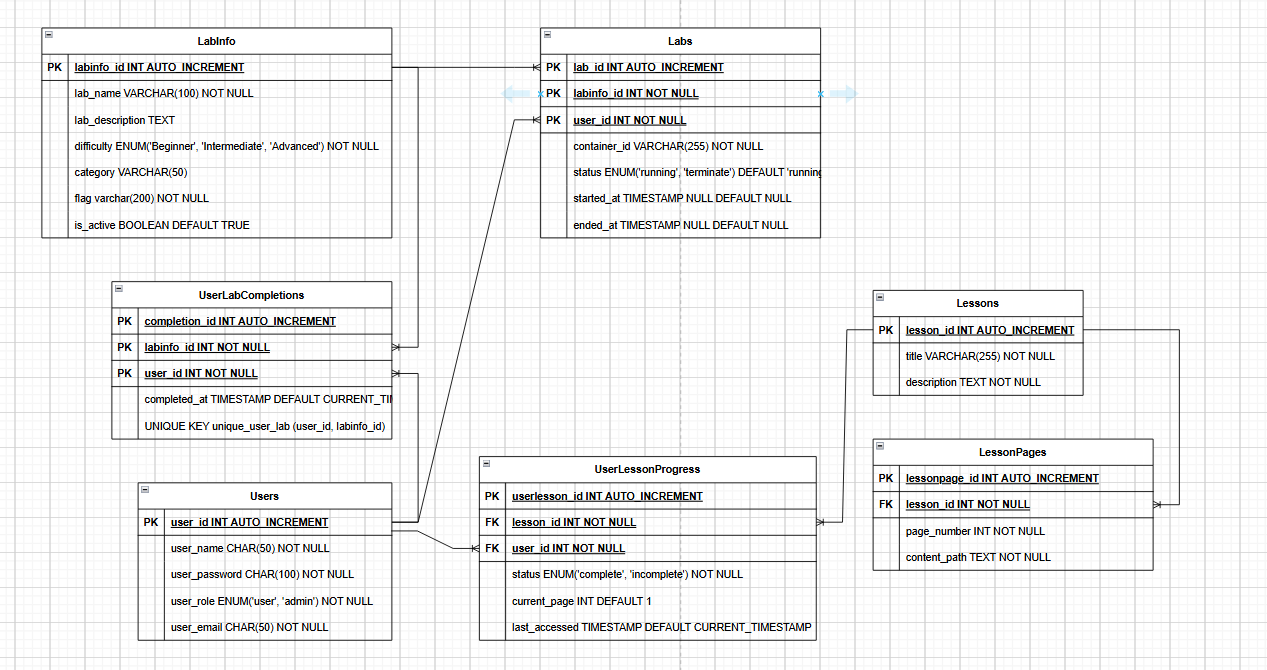
* **lessonService**: Facilitates CRUD operations for lessons, content retrieval, and progress tracking
* **labService**: Manages interactive lab environments, exercise submissions, and progress evaluation

#### Service Architecture Patterns

* Implementation of the repository pattern for data access abstraction
* Consistent error handling and response normalization
* Caching strategies for frequently accessed data
* Request interceptors for authentication header injection
* Response transformation for component-ready data structures

## Database design

### Entity-Relationship Diagram (ERD)



#### Users table

Users(user\_id, user\_name, user\_password, user\_role, user\_email)

#### LabInfo table

LabInfo(labinfo\_id, lab\_name, lab\_description, difficulty, category, flag, is\_active)

#### Labs table

Labs(lab\_id, user\_id, labinfo\_id, container\_id, status, started\_at, ended\_at)

from Labs.user\_id to Users.user\_id

from Labs.labinfo\_id to LabInfo.labinfo\_id

#### UserLabCompletions table

UserLabCompletions(completion\_id, user\_id, labinfo\_id, completed\_at)

from UserLabCompletions.user\_id to Users.user\_id

from UserLabCompletions.labinfo\_id to LabInfo.labinfo\_id

#### Lessons table

Lessons(lesson\_id, title, description)

#### LessonPages table

LessonPages(lessonpage\_id, lesson\_id, page\_number, content\_path)

from LessonPages.lesson\_id to Lessons.lesson\_id

#### UserLessonProgress table

UserLessonProgress(userlesson\_id, user\_id, lesson\_id, status, current\_page, last\_accessed)

from UserLessonProgress.user\_id to Users.user\_id

from UserLessonProgress.lesson\_id to Lessons.lesson\_id

## Use Case Diagram

A diagram of a learning process

AI-generated content may be incorrect.

### Use case 1: Log in

A diagram of a software project

AI-generated content may be incorrect.

### Use case 2: Register

A diagram of a graph

AI-generated content may be incorrect.

### Use case 3: Browsing Lesson

A diagram of a computer

AI-generated content may be incorrect.

### Use case 4: Completing Lesson

A diagram of a project

AI-generated content may be incorrect.

### Use case 5: Browsing lab

A diagram of a computer program

AI-generated content may be incorrect.

### Use case 6: Starting a lab

A diagram of a project

AI-generated content may be incorrect.

### Use case 7: Submiting a flag

A diagram of a diagram

AI-generated content may be incorrect.

### Use case 8: Stopping a lab

A diagram of a company

AI-generated content may be incorrect.

### Use case 9: Managing Account

A diagram of a computer program

AI-generated content may be incorrect.

### Use case 10: Create Lesson

A diagram of a diagram

AI-generated content may be incorrect.

### Use case 11: Edit Lesson

A diagram of a company

AI-generated content may be incorrect.

# Implementation

User’s

# Discussion and conclusion

**Question 1:** What is the difference between the function of graduated and volumetric pipette?

Quisque at sapien turpis. Etiam libero tellus, ultrices et nibh sit amet, fermentum dignissim lorem. Nullam malesuada tortor sed volutpat egestas. Fusce id justo gravida, maximus elit in, vulputate ipsum. In sapien tortor, varius eget neque nec, ornare porttitor sem. In mattis nec nibh et finibus. Etiam placerat velit vel nisl sodales viverra.

**Question 2:** What is the difference between the function of graduated and volumetric pipette?

Quisque at sapien turpis. Etiam libero tellus, ultrices et nibh sit amet, fermentum dignissim lorem. Nullam malesuada tortor sed volutpat egestas. Fusce id justo gravida, maximus elit in, vulputate ipsum. In sapien tortor, varius eget neque nec, ornare porttitor sem. In mattis nec nibh et finibus. Etiam placerat velit vel nisl sodales viverra.

# Conclusion

Morbi leo justo, ultrices luctus dignissim ut, semper ac velit. Ut quam nunc, tempus eu blandit eget, scelerisque vitae odio. Fusce euismod magna ut sollicitudin auctor. Maecenas mollis vestibulum urna, eget feugiat nibh condimentum sed. Cras commodo, enim eu hendrerit congue, nulla ipsum commodo enim, id interdum magna ex sit amet mi. Etiam aliquet, lorem a commodo sodales, ante odio fermentum diam, vel placerat felis purus eu tortor. Duis volutpat scelerisque augue, sed pellentesque purus faucibus sit amet. Aliquam viverra massa in mi placerat, nec facilisis nisi cursus. Phasellus ac elit eu orci euismod sodales id ac lectus. Duis in posuere nibh. Suspendisse aliquam convallis nunc vitae condimentum.

# References

*(Please make sure that the references should be in APA format and should be have in-text citations. You can use EndNote, Mendeley or any citation machine that available. To aquire more information, access* [*https://www.mendeley.com/guides/apa-citation-guide/*](https://www.mendeley.com/guides/apa-citation-guide/)*)*