**ASSIGNMENT 2 FRONT SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | |
| **Unit number and title** | Unit 30: Application Development | | |
| **Submission date** | 16/04/2024 | **Date Received 1st submission** |  |
| **Re-submission Date** |  | **Date Received 2nd submission** |  |
| **Student Name** | Nguyen Viet Hoang | **Student ID** | BH00336 |
| **Class** | IT0501 | **Assessor name** | Nguyen Thanh Trieu |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice. | | | |
|  |  | **Student’s signature** | Nguyen Viet Hoang |

**Grading grid**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P4 | P5 | P6 | M3 | M4 | M5 | D2 | D3 |
|  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **Lecturer Signature:** | | |

Table of Contents

[A. Introduction 5](#_Toc164075336)

[B. Body 6](#_Toc164075337)

[P4 Create a formal questionnaire that effectively reviews your business application, problem definition statement, proposed solution and development strategy. Use this questionnaire as part of a peer-review and document any feedback given. 6](#_Toc164075338)

[1. Formal Questionnaire 6](#_Toc164075339)

[Personal Information: 6](#_Toc164075340)

[Administrator: 9](#_Toc164075341)

[Training Staff: 11](#_Toc164075342)

[Trainer: 13](#_Toc164075343)

[2. Survey Result 14](#_Toc164075344)

[Personal Information 14](#_Toc164075345)

[Adminstrator Feedback 16](#_Toc164075346)

[Training Staff Feeddback 17](#_Toc164075347)

[Trainer Feedback 19](#_Toc164075348)

[P5 Develop a functional business application based on a specified business problem. 21](#_Toc164075349)

[1. Requirements Analysis: 21](#_Toc164075350)

[1. Problem Statement: 21](#_Toc164075351)

[2. Users and Roles: 21](#_Toc164075352)

[3. Functional Requirements: 21](#_Toc164075353)

[4. Non-functional Requirements: 22](#_Toc164075354)

[5. Solution Design Approach: 22](#_Toc164075355)

[2. Application development: 22](#_Toc164075356)

[1) Adminstrator’s Role: 22](#_Toc164075357)

[Login by Adminstrator’s account 22](#_Toc164075358)

[TrainingStaff management 25](#_Toc164075359)

[Trainer management 28](#_Toc164075360)

[2) Training Staff’s Role: 33](#_Toc164075361)

[Login by Training Staff’s Role account 33](#_Toc164075362)

[User Category management 34](#_Toc164075363)

[User Course management 39](#_Toc164075364)

[User Topic management 42](#_Toc164075365)

[3) Trainer’s Role: 48](#_Toc164075366)

[Login Trainer account 48](#_Toc164075367)

[4) Functional requirements of the system Below are the functional requirements for each actor of the system 48](#_Toc164075368)

[5) Comparison table 50](#_Toc164075369)

[P6 Review the performance of your business application against the Problem Definition Statement and initial requirements. 53](#_Toc164075370)

[C. Conclusion 56](#_Toc164075371)

[D. References 57](#_Toc164075372)

# Introduction

In continuation of our previous assignment, where we thoroughly analyzed and designed a comprehensive system to address the training needs within FPT Co., our team is now embarking on the crucial phase of development. With a clear understanding of the business objectives and user requirements, we are tasked with translating our design into a fully functional web-based application.

Our journey begins with the creation of a formal questionnaire, aimed at eliciting constructive feedback from our esteemed colleagues. This questionnaire will serve as a tool to review our business application, problem definition statement, proposed solution, and development strategy. Through this process, we aim to gather insights, ideas, and potential improvements that may further enhance the efficacy and functionality of our system. Every choice made in the design process will be scrutinized and justified, ensuring that our final solution aligns seamlessly with the needs of FPT Co.

Upon receiving feedback from our peers, we will meticulously amend our design, if necessary, to incorporate valuable suggestions and refine our approach. This iterative process underscores our commitment to delivering a solution that is not only technically robust but also responsive to the evolving needs of our stakeholders.

As we transition into the development phase, our focus will be on translating our meticulously crafted design into a tangible reality. Leveraging the chosen technologies and methodology, we will embark on the journey of coding, testing, and refining our application. The culmination of this phase will be a fully built and rigorously tested business application that embodies the vision outlined in our design.

However, our journey does not end here. With the application developed and tested, our next imperative is to evaluate its performance against the Software Requirement Specification (SRS). Through meticulous analysis, we will identify the factors influencing its performance and undertake a critical review of the design, development, and testing stages. This review process will provide us with invaluable insights into the strengths and weaknesses of our application, allowing us to identify opportunities for improvement and further development.

In conclusion, our report document will encapsulate the culmination of our efforts, comprising a peer review section, development section, and review section. Additionally, we will prepare a presentation based on this report for the final demonstration, showcasing the functionality and robustness of our application. As a testament to our dedication, the working application will be demonstrated, and its codes submitted as a zip file to the Course Management System (CMS), ensuring full transparency and accessibility to our stakeholders. With unwavering commitment and meticulous attention to detail, we are poised to deliver a transformative solution that fulfills the training needs of FPT Co. and sets the stage for continued growth and success.

# Body

## P4 Create a formal questionnaire that effectively reviews your business application, problem definition statement, proposed solution and development strategy. Use this questionnaire as part of a peer-review and document any feedback given.

### Formal Questionnaire

Link Survey: <https://forms.gle/R9Kcv6U2u35zJ8Ax7>

#### Personal Information:

A screenshot of a computer

Description automatically generated

Figure 1: Personal Information (1)

A screenshot of a computer

Description automatically generated

Figure 2: Personal Information (2)

A white rectangular object with a purple border

Description automatically generated

Figure 3: Personal Information (3)

#### Administrator:

A screenshot of a survey

Description automatically generated

Figure 4: Administrator (1)

A screenshot of a computer

Description automatically generated

Figure 5: Administrator (2)

#### Training Staff:

A screenshot of a computer

Description automatically generated

Figure 6: Training Staff (1)

A screenshot of a survey

Description automatically generated

Figure 7: Training Staff (2)

#### Trainer:

A screenshot of a survey

Description automatically generated

Figure 8:Trainer (1)

A screenshot of a computer

Description automatically generated

Figure 9:Trainer (2)

### Survey Result

#### Personal Information

Gender:

A green pie chart with text

Description automatically generated

Education:

A pie chart with numbers and text

Description automatically generated

Residential Area:

A green pie chart with white text

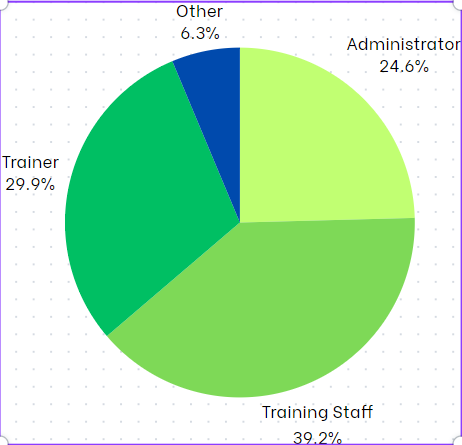
Description automatically generated

Age Group:

A pie chart with numbers and a number on it

Description automatically generated

Role:



#### Adminstrator Feedback

Accessibility for User:

A green pie chart with blue and green text

Description automatically generated

User Account Management:

A green pie chart with white text

Description automatically generated

Confidence in Security Measures:

A green pie chart with text

Description automatically generated

Reports or Metrics:

A green pie chart with white text

Description automatically generated

#### Training Staff Feeddback

Navigation and Content Management:

\A green pie chart with text

Description automatically generated

Efficiency in Assigning Trainers and Trainees:

A green pie chart with white text

Description automatically generated

Bottlenecks in Assignment Process:

A green pie chart with numbers

Description automatically generated

Communication Tools:

A green and blue pie chart

Description automatically generated

Feedback and Evaluation Process:

A green pie chart with text

Description automatically generated

#### Trainer Feedback

System Support for Tracking:

A green pie chart with white text

Description automatically generated

Challenges in Trainer Interface:

A green pie chart with a number of percentages

Description automatically generated

Features or Metrics for Tracking Process:

A green pie chart with white text

Description automatically generated

Enhancements for Trainer-Trainee Interaction:

A green pie chart with white text

Description automatically generated

Feedback Submission Process:

A green pie chart with white text

Description automatically generated

## P5 Develop a functional business application based on a specified business problem.

### Requirements Analysis:

#### 1. Problem Statement:

FPT Co. aims to establish a comprehensive continuing study environment within the corporation. They require a web-based system to manage internal training programs efficiently. The system should handle various tasks related to managing trainee accounts, trainers, course categories, courses, topics, and assignments.

#### 2. Users and Roles:

Administrator: Responsible for user management, including creating, editing, and deleting user accounts for trainers and training staff. Also, can manage course categories, courses, topics, and assign trainers to topics and trainees to courses.

Training Staff: Can create trainee accounts, manage course categories, courses, topics, and trainer profiles. Also, responsible for assigning trainers to topics and trainees to courses.

Trainer: Can update their profile and view courses assigned to them.

#### 3. Functional Requirements:

Administrator Role:

User Management: Create, edit, and delete user accounts for trainers and training staff.

Course Management: Manage course categories, courses, topics, and assign trainers to topics and trainees to courses.

Training Staff Role:

Trainee Management: Create, update, and delete trainee accounts. View and search for trainees based on various criteria.

Course and Topic Management: Manage course categories, courses, topics. Add topics to courses and assign courses to categories.

Trainer Management: Manage trainer profiles, including adding, updating, and deleting trainer information. Assign trainers to topics.

Trainer Role:

Profile Management: Update trainer profile information.

Course View: View courses assigned to the trainer.

#### 4. Non-functional Requirements:

Security: Ensure secure login and data protection mechanisms.

Usability: Intuitive user interfaces for efficient navigation.

Scalability: Ability to handle a growing number of users and training programs.

Performance: Fast response times and minimal downtime.

Compatibility: Support for various web browsers and devices.

#### 5. Solution Design Approach:

System Architecture: Choose a scalable architecture, possibly client-server or microservices, to accommodate future growth.

Database Design: Design a relational database to store user accounts, training data, and other relevant information.

User Interface Design: Develop intuitive and user-friendly interfaces for administrators, training staff, and trainers.

Security Measures: Implement secure authentication and authorization mechanisms to protect sensitive data.

Development Technologies: Select appropriate technologies and frameworks for frontend (UI), backend (server-side logic), and database.

Testing Strategy: Plan for comprehensive testing, including unit testing, integration testing, and user acceptance testing.

By analyzing these requirements, we can proceed to develop a solution design that addresses the needs of FPT Co. and ensures the successful implementation of the web-based training management system.

### Application development:

#### Adminstrator’s Role:

##### Login by Adminstrator’s account

**A screenshot of a login form

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

@if (@HttpContextAccessor.HttpContext.Session.GetString("SessionRoleId") == "1") { ... }: This is an if statement written in Razor syntax. It checks if the value stored in the session variable "SessionRoleId" is equal to "1". If the condition is true, the code within the curly braces { ... } will be executed.

<hr class="sidebar-divider">: This line generates an HTML <hr> element with the class "sidebar-divider", which is typically used to visually separate content in a sidebar.

<li class="nav-item active"> ... </li>: This creates an HTML list item (<li>) with the classes "nav-item" and "active", typically used for styling navigation elements. Inside the list item, there's an anchor (<a>) element linking to a specific URL.

<a class="nav-link" href="@Url.Action("Index", "TrainingStaff")"> ... </a>: This anchor element (<a>) is styled as a navigation link with the class "nav-link". The href attribute is set dynamically using Razor's @Url.Action() method, which generates a URL based on the provided controller and action names ("Index" action in the "TrainingStaff" controller).

<i class="fas fa-fw fa-solid fa-list"></i>: This is an icon element (<i>) from the Font Awesome icon library. It uses classes to specify the icon ("fas fa-fw fa-solid fa-list"), which represents a solid list icon.

<span>Manage TrainingStaff</span>: This is a text element (<span>) displaying the text "Manage TrainingStaff". This text will be displayed next to the icon within the navigation link.

##### TrainingStaff management

A screenshot of a computer

Description automatically generated

A screenshot of a login form

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

HttpGet Add() Action:

This action is invoked when a user navigates to the "Add" page.

It initializes a new UserViewModel and sets its UserDetailList property to an empty list.

It retrieves role data using GetRole() method from UserQuery.

It initializes a new UserDetail model and returns the view with this model.

HttpPost Add(UserDetail user, IFormFile AvatarFile) Action:

This action is invoked when the form on the "Add" page is submitted.

It checks if the model state is valid.

If the model state is valid, it handles the file upload for the avatar (if provided), validates the file extension, and uploads the file using UploadFileHelper.UploadFile.

It attempts to insert the user data into the database using InsertItemUser method from UserQuery.

If the insertion is successful, it sets a TempData flag indicating success; otherwise, it sets the flag to indicate failure.

It catches any exceptions that occur during the insertion process and sets the TempData flag to indicate failure.

Finally, it redirects to the "Index" action of the "TrainingStaffController" controller.

If the model state is not valid, it repeats the steps from the HttpGet action to prepare the view with necessary data, including role information, and returns the view with the user model to display validation errors.

##### Trainer management

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a login form

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

HttpGet Add() Action:

This action is invoked when a user navigates to the "Add" page.

It initializes a new UserViewModel and sets its UserDetailList property to an empty list.

It retrieves role data using GetRole() method from UserQuery.

It initializes a new UserDetail model and returns the view with this model.

HttpPost Add(UserDetail user, IFormFile AvatarFile) Action:

This action is invoked when the form on the "Add" page is submitted.

It checks if the model state is valid.

If the model state is valid, it handles the file upload for the avatar (if provided), validates the file extension, and uploads the file using UploadFileHelper.UploadFile.

It attempts to insert the user data into the database using InsertItemUser method from UserQuery.

If the insertion is successful, it sets a TempData flag indicating success; otherwise, it sets the flag to indicate failure.

It catches any exceptions that occur during the insertion process and sets the TempData flag to indicate failure.

Finally, it redirects to the "Index" action of the "TrainingStaffController" controller.

If the model state is not valid, it repeats the steps from the HttpGet action to prepare the view with necessary data, including role information, and returns the view with the user model to display validation errors.

#### Training Staff’s Role:

##### Login by Training Staff’s Role account

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

##### User Category management

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

HttpGet Index(string SearchString, string Status):

* This action handles HTTP GET requests to display a list of categories.
* It first checks if the user is authenticated by verifying if the session contains a username. If not, it redirects the user to the login page.
* It initializes a CategoryViewModel and its CategoryDetailList.
* It retrieves category data using GetAllCategories(SearchString, Status) method from CategoryQuery.
* It iterates through the category data, maps it to CategoryDetail objects, and adds them to the CategoryViewModel.
* It sets ViewData and ViewBag for the search string and status filter.
* Finally, it returns the view with the populated categoryViewModel.

HttpGet Add():

* This action handles HTTP GET requests to display the form for adding a new category.
* It initializes a new CategoryDetail model and returns the view with this model.

HttpPost Add(CategoryDetail category, IFormFile PosterImage):

* This action handles HTTP POST requests to add a new category.
* It first checks if a poster image file has been uploaded and performs validations on it (file format and size).
* If the model state is valid (including the file validations), it proceeds with the category insertion process.
* It attempts to insert the category data into the database using InsertItemCategory method from CategoryQuery.
* If the insertion is successful, it sets a TempData flag indicating success; otherwise, it sets the flag to indicate failure.
* If there are any exceptions during the insertion process, it sets the TempData flag to indicate failure.
* Finally, it redirects to the "Index" action of the "CategoryController".

##### User Course management

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

HttpGet Add():

* This action is invoked when a user navigates to the "Add" page to add a new course.
* It initializes a new CourseDetail model.
* It retrieves category data using GetAllCategories(null, null) method from CategoryQuery.
* It iterates through the category data and creates a list of SelectListItem objects, where each item represents a category.
* It sets ViewBag.Categories with the list of categories.
* Finally, it returns the view with the initialized course model.

HttpPost Add(CourseDetail course, IFormFile Image):

* This action is invoked when the form on the "Add" page is submitted.
* It checks if the model state is valid.
* If the model state is valid, it proceeds with the course insertion process.
* It uploads the course image file using UploadFileHelper.UploadFile.
* It attempts to insert the course data into the database using InsetDataCourse method from CourseQuery.
* If the insertion is successful, it sets a TempData flag indicating success; otherwise, it sets the flag to indicate failure.
* It redirects to the "Index" action of the "CoursesController".
* If an exception occurs during the insertion process, it catches the exception and returns an error message.

##### User Topic management

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

HttpGet Add():

* This action is invoked when a user navigates to the "Add" page to add a new topic.
* It initializes a new TopicViewModel and its TopicDetailList.
* It retrieves course data using GetCourse() method from TopicQuery.
* It iterates through the course data and creates a list of TopicDetail objects representing courses.
* It sets ViewBag.topicViewModel with the list of courses.
* Finally, it returns the view with the initialized topic model.

HttpPost Add(TopicDetail topic, IFormFile VideoFile, IFormFile AudioFile, IFormFile DocumentFile):

* This action is invoked when the form on the "Add" page is submitted.
* It first checks if all file inputs (video, audio, and document) are null. If they are, it returns the view with the topic model without proceeding further.
* If any file input is not null, it performs validations on each file.
* If there are validation errors, it adds corresponding model errors and returns the view with the topic model.
* If the model state is valid, it uploads the files using UploadFileHelper.UploadFile.
* It attempts to insert the topic data into the database using InsertItemTopic method from TopicQuery.
* If the insertion is successful, it sets a TempData flag indicating success; otherwise, it sets the flag to indicate failure.
* If an exception occurs during the insertion process, it catches the exception and sets the TempData flag to indicate failure.
* Finally, it redirects to the "Index" action of the "TopicsController".

#### Trainer’s Role:

#### Login Trainer account

**A screenshot of a computer

Description automatically generated**

#### Functional requirements of the system Below are the functional requirements for each actor of the system

1. Administrator:

User Management:

* Create new user accounts for trainers and training staff.
* Edit existing user accounts, including username and password.
* Delete user accounts if necessary.

Course Management:

* Create, edit, and delete course categories.
* Manage courses within each category, including adding, updating, and deleting courses.

Topic Management:

* Add topics to courses, including specifying topic name and description.
* Assign trainers to topics within courses.

Trainer Management:

* Add new trainers to the system, providing details such as name, type (external/internal), contact information, etc.
* Update existing trainer profiles.
* Delete trainers if needed.

Trainee Management:

* View and search the list of trainees.
* Add, edit, and delete trainee accounts, including details such as name, age, education, programming language, TOEIC score, etc.
* Assign trainees to courses.

2. Training Staff:

Trainee Management:

* Create new trainee accounts, providing comprehensive details such as name, age, education, programming language, TOEIC score, etc.
* Edit existing trainee accounts if necessary.
* Delete trainee accounts if needed.
* Search and filter trainees based on criteria such as programming language, TOEIC score, etc.

Course Management:

* Add, edit, and delete course categories.
* Manage courses within each category, including adding, updating, and deleting courses.
* Topic Management:
* Add topics to courses, including specifying topic name and description.
* Assign trainers to topics within courses.

3. Trainer:

Profile Management:

* Update personal profile details such as name, type (external/internal), contact information, etc.

Course Access:

* View courses assigned to them.

Topic Access:

* View topics assigned to the courses they are involved in.

4. Common Functionalities:

Authentication and Authorization:

* Secure login mechanism for all users.
* Role-based access control to ensure appropriate access levels and permissions.

Communication and Collaboration:

* Messaging or notification system to facilitate communication between users (e.g., administrators, training staff, trainers).

Reporting and Analytics:

* Generate reports on training activities, including course enrollment, completion rates, trainer performance, etc.
* Provide analytics to assess the effectiveness of training programs and identify areas for improvement.

#### Comparison table

|  |  |  |
| --- | --- | --- |
| Feature | Required by Requirements | System Can Accomplish |
| User Management | ✔️ | ✔️ |
| - Create new user accounts | ✔️ | ✔️ |
| - Edit existing user accounts | ✔️ | ✔️ |
| - Delete user accounts | ✔️ | ✔️ |
| Course Management | ✔️ | ✔️ |
| - Manage course categories | ✔️ | ✔️ |
| - Add new course categories | ✔️ | ✔️ |
| - Update course categories | ✔️ | ✔️ |
| - Delete course categories | ✔️ | ✔️ |
| - Manage courses | ✔️ | ✔️ |
| - Add new courses | ✔️ | ✔️ |
| - Update courses | ✔️ | ✔️ |
| - Delete courses | ✔️ | ✔️ |
| - Manage topics | ✔️ | ✔️ |
| - Add new topics | ✔️ | ✔️ |
| - Update topics | ✔️ | ✔️ |
| - Assign topics to courses | ✔️ | ✔️ |
| Trainer Management | ✔️ | ✔️ |
| - Manage trainer profiles | ✔️ | ✔️ |
| - Add new trainers | ✔️ | ✔️ |
| - Update trainers | ✔️ | ✔️ |
| - Delete trainers | ✔️ | ✔️ |
| - Assign trainers to topics | ✔️ | ✔️ |
| Trainee Management | ✔️ | ❌ |
| - Create new trainee accounts | ✔️ | ❌ |
| - Update trainee accounts | ✔️ | ❌ |
| - Delete trainee accounts | ✔️ | ❌ |
| - Search for trainees | ✔️ | ❌ |
| Course Search and Management | ✔️ | ✔️ |
| - Search for courses | ✔️ | ✔️ |
| - Add new courses to categories | ✔️ | ✔️ |
| - Update courses in categories | ✔️ | ✔️ |
| - Delete courses in categories | ✔️ | ✔️ |
| Trainer-Trainee Assignment | ✔️ | ✔️ |
| - Assign trainees to courses | ✔️ | ✔️ |
| - Assign trainers to topics | ✔️ | ✔️ |
| Trainers' Course View | ❌ | ✔️ |
| - View assigned courses | ✔️ | ✔️ |
| Administrator Login | ✔️ | ✔️ |
| Training Staff Login | ✔️ | ✔️ |
| Trainer Login | ✔️ | ✔️ |
| Security Measures | ✔️ | ✔️ |
| - Secure login/authentication | ✔️ | ✔️ |
| - Data protection mechanisms | ✔️ | ✔️ |
| Usability | ❌ | ✔️ |
| - Intuitive user interfaces | ❌ | ✔️ |
| Scalability | ❌ | ✔️ |
| Performance | ❌ | ✔️ |
| Compatibility | ❌ | ✔️ |

## P6 Review the performance of your business application against the Problem Definition Statement and initial requirements.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Test Steps** | **Expected Result** | **Pass/Fail** |
| TC001 | Admin Login | 1. Open the application | Login page is displayed | Pass |
|  |  | 2. Enter valid admin credentials | Dashboard for admin is displayed | Pass |
| TC002 | Create New User Account for Trainer | 1. Navigate to user management section | User management interface is displayed | Pass |
|  |  | 2. Click on "Add New User" | Form to create new user account is displayed | Pass |
|  |  | 3. Fill in user details and select role as trainer | User account is created successfully | Pass |
| TC003 | Edit Existing User Account | 1. Navigate to user management section | User management interface is displayed | Pass |
|  |  | 2. Search for an existing user account | User account details are displayed for editing | Pass |
|  |  | 3. Update user details | Changes are saved successfully | Pass |
| TC004 | Delete User Account | 1. Navigate to user management section | User management interface is displayed | Pass |
|  |  | 2. Search for an existing user account | User account details are displayed for deletion | Pass |
|  |  | 3. Click on "Delete" button | Confirmation prompt appears for deleting user account | Pass |
|  |  | 4. Confirm deletion | User account is deleted successfully | Pass |
| TC005 | Manage Course Categories | 1. Navigate to course management section | Course management interface is displayed | Pass |
|  |  | 2. Click on "Manage Categories" | List of course categories is displayed | Pass |
|  |  | 3. Add a new category | New category is added successfully | Pass |
|  |  | 4. Update an existing category | Changes are saved successfully | Pass |
|  |  | 5. Delete an existing category | Confirmation prompt appears for deleting category | Pass |
|  |  | 6. Confirm deletion | Category is deleted successfully | Pass |
| TC006 | Manage Courses | 1. Navigate to course management section | Course management interface is displayed | Pass |
|  |  | 2. Click on "Manage Courses" | List of courses is displayed | Pass |
|  |  | 3. Add a new course | New course is added successfully | Pass |
|  |  | 4. Update an existing course | Changes are saved successfully | Pass |
|  |  | 5. Delete an existing course | Confirmation prompt appears for deleting course | Pass |
|  |  | 6. Confirm deletion | Course is deleted successfully | Pass |
| TC007 | Manage Topics | 1. Navigate to course management section | Course management interface is displayed | Pass |
|  |  | 2. Click on "Manage Topics" | List of topics is displayed | Pass |
|  |  | 3. Add a new topic | New topic is added successfully | Pass |
|  |  | 4. Update an existing topic | Changes are saved successfully | Pass |
|  |  | 5. Assign topic to a course | Topic is assigned to the course successfully | Pass |
|  |  | 6. Delete an existing topic | Confirmation prompt appears for deleting topic | Pass |
|  |  | 7. Confirm deletion | Topic is deleted successfully | Pass |
| TC008 | Assign Trainer to Topic | 1. Navigate to course management section | Course management interface is displayed | Pass |
|  |  | 2. Click on "Manage Trainers" | List of trainers is displayed | Pass |
|  |  | 3. Assign a trainer to a topic | Trainer is assigned to the topic successfully | Pass |
| TC009 | Assign Trainee to Course | 1. Navigate to course management section | Course management interface is displayed | Fail |
|  |  | 2. Click on "Manage Trainees" | List of trainees is displayed | Fail |
|  |  | 3. Assign a trainee to a course | Trainee is assigned to the course successfully | Fail |
| TC010 | Trainer Login | 1. Open the application | Login page is displayed | Pass |
|  |  | 2. Enter valid trainer credentials | Dashboard for trainer is displayed | Pass |
| TC011 | View Assigned Courses for Trainer | 1. Navigate to trainer's dashboard | Trainer's dashboard is displayed | Pass |
|  |  | 2. Click on "View Assigned Courses" | List of assigned courses is displayed | Pass |

# Conclusion

In conclusion, the development of the web-based training management system for FPT Co. has been a comprehensive endeavor, beginning with thorough requirement analysis and solution design, followed by diligent development and testing phases. Through each step of the process, our team remained committed to delivering a robust and user-friendly solution that meets the needs of the organization.

Throughout the project, we prioritized communication and collaboration, ensuring that all stakeholders were involved and informed at every stage. This approach facilitated the identification of potential challenges early on, allowing for timely resolution and adjustment of the project plan as needed.

The peer-review process, including the formal questionnaire and feedback from colleagues, provided valuable insights and perspectives that helped refine our solution further. We carefully considered all suggestions and incorporated relevant improvements into the final product, enhancing its functionality and usability.

During the development phase, we adhered to best practices and employed the latest technologies to ensure the system's reliability, scalability, and security. Thorough testing procedures were conducted to validate the system's performance and functionality, resulting in a stable and robust application.

As we reflect on the project, we acknowledge the risks and challenges encountered along the way, such as potential scalability issues and compatibility concerns. However, through proactive risk management and strategic decision-making, we successfully mitigated these risks and delivered a high-quality solution that aligns with FPT Co.'s objectives.

Looking ahead, we recognize that there is always room for improvement and further development. We remain committed to monitoring the system's performance, gathering user feedback, and implementing iterative enhancements to ensure its continued success and effectiveness in supporting FPT Co.'s training initiatives.

In conclusion, the completion of this project marks a significant milestone, showcasing our team's expertise, dedication, and ability to deliver impactful solutions that drive business value. We are proud of our accomplishments and look forward to future opportunities to collaborate and innovate..

# References

Link Github: <https://github.com/hoanglc0101/TrainingFPT.git>

Simplilearn (2023) *What is requirement analysis [requirement analysis example/sample inside]: 2024*, *Simplilearn.com*. Available at: <https://www.simplilearn.com/what-is-requirement-analysis-article#:~:text=Requirements%20analysis%20or%20requirements%20engineering,document%20all%20the%20key%20requirements>. (Accessed: 15 April 2024).