**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Examination Tools applying Block Chain Technology**

|  |  |
| --- | --- |
| Group 4 | |
| Group members | Nguyễn Minh Hưng – SE61794  Cao Trung Hiếu – SE61801  Trương Tấn Sang – SE61926 |
| Supervisor | Kiều Trọng Khánh |
| Ext. Supervisor | N/A |
| Capstone Project code | ExamTool |

- Ho Chi Minh City, **11 September 2018** -

Table of Contents

[Table of Contents 2](#_Toc531275980)

[List Of Tables 8](#_Toc531275981)

[List Of Figures 9](#_Toc531275982)

[Definitions, Acronyms, and Abbreviations 10](#_Toc531275983)

[A. Introduction 11](#_Toc531275984)

[1. Project Information 11](#_Toc531275985)

[2. Introduction 11](#_Toc531275986)

[3. Current Situation 11](#_Toc531275987)

[4. Problem Definition 12](#_Toc531275988)

[4.1. Proposed Solution 12](#_Toc531275989)

[4.2. Feature Functions 12](#_Toc531275990)

[5. Values and Challenges Value 12](#_Toc531275991)

[6. Functional Requirement 12](#_Toc531275992)

[7. Role and Responsibility 13](#_Toc531275993)

[B. Software Project Management Plan 14](#_Toc531275994)

[1. Problem Definition 14](#_Toc531275995)

[1.1. Name of this Capstone Project 14](#_Toc531275996)

[1.2. Problem Abstract 14](#_Toc531275997)

[1.3. Project Overview 14](#_Toc531275998)

[1.3.1 Current Situation 14](#_Toc531275999)

[1.3.2 Boundaries of the system 14](#_Toc531276000)

[1.3.3 Future Plans 15](#_Toc531276001)

[1.3.4 Development Environment 15](#_Toc531276002)

[1.3.4.1 Hardware requirements 15](#_Toc531276003)

[1.3.4.2 Software requirements 15](#_Toc531276004)

[2. Coding Convention 15](#_Toc531276005)

[3. Project Organization 16](#_Toc531276006)

[3.1. Software Process Model 16](#_Toc531276007)

[3.2. Roles and Responsibilities 17](#_Toc531276008)

[3.3. Tools and Techniques 18](#_Toc531276009)

[4. Project Management Plan 18](#_Toc531276010)

[4.1. Product Backlog 18](#_Toc531276011)

[4.2. Sprint Backlog 19](#_Toc531276012)

[4.2.1 Sprint 1 (10.09.2018 – 23.09.2018): Project initiation 19](#_Toc531276013)

[4.2.1.1 Goal 19](#_Toc531276014)

[4.2.1.2 Development 19](#_Toc531276015)

[4.2.2 Sprint 2 (24.09.2018 – 8.10.2018): Software Document 20](#_Toc531276016)

[4.2.2.1 Goal 20](#_Toc531276017)

[4.2.2.2 Development 20](#_Toc531276018)

[4.2.3 Sprint 3 (9.10.2018 – 23.10.2018): Basic feature and core 21](#_Toc531276019)

[4.2.3.1 Goal 21](#_Toc531276020)

[4.2.3.2 Development 21](#_Toc531276021)

[4.2.4 Sprint 4 (24.10.2018 – 8.11.2018): Software Document 21](#_Toc531276022)

[4.2.4.1 Goal 21](#_Toc531276023)

[4.2.4.2 Development 22](#_Toc531276024)

[4.2.5 Sprint 5 (9.11.2018 – 23.11.2018): Software Document 22](#_Toc531276025)

[4.2.5.1 Goal 22](#_Toc531276026)

[4.2.5.2 Development 23](#_Toc531276027)

[4.2.6 Sprint 6 (24.11.2018 – 13.12.2018): Complete Document 23](#_Toc531276028)

[4.2.6.1 Goal 23](#_Toc531276029)

[4.2.6.2 Development 23](#_Toc531276030)

[C. Software Requirement Specification 24](#_Toc531276031)

[1. User Requirement Specification 24](#_Toc531276032)

[1.1. Examination Requirement 24](#_Toc531276033)

[1.2. System Requirement 24](#_Toc531276034)

[1.3. Administrator Requirement 24](#_Toc531276035)

[2. System Requirement Specification 24](#_Toc531276036)

[2.1. External Interface Requirement 24](#_Toc531276037)

[2.1.1 User Interface 24](#_Toc531276038)

[2.1.2 Hardware Interface 24](#_Toc531276039)

[2.1.3 Software Interface 24](#_Toc531276040)

[2.1.4 Communication Protocol 24](#_Toc531276041)

[2.2. System Overview Diagram 25](#_Toc531276042)

[2.3. List of use case 29](#_Toc531276043)

[2.3.1 <IT> Get List Process Must Kill 29](#_Toc531276044)

[2.3.2 <IT> Remove Process Must be Killed 30](#_Toc531276045)

[2.3.3 <IT> Add Must Be Killed Process to List 31](#_Toc531276046)

[2.3.4 <IT> Add to List Exception Process 32](#_Toc531276047)

[2.3.5 <IT> Kill Process 33](#_Toc531276048)

[2.3.6 <IT> Get Realtime Process 35](#_Toc531276049)

[2.3.7 <IT> Search Realtime Process 36](#_Toc531276050)

[2.3.8 <IT> Get Student Screen 37](#_Toc531276051)

[2.3.9 <IT> Submit Student Test Exam 39](#_Toc531276052)

[2.3.10 <IT> Search Online Student 40](#_Toc531276053)

[2.3.11 <Training Department Staff> Add new Semester 42](#_Toc531276054)

[2.3.12 <Training Department Staff> Get Semester Detail 44](#_Toc531276055)

[2.3.13 <Manager> Approve Available Test Exam 45](#_Toc531276056)

[2.3.14 <Leader> Search Chapter 47](#_Toc531276057)

[2.3.15 <Leader> Add chapter 48](#_Toc531276058)

[2.3.16 <Leader> Delete chapter 51](#_Toc531276059)

[2.3.17 <Leader> Edit chapter 53](#_Toc531276060)

[2.3.18 <Teacher> Search Question 55](#_Toc531276061)

[2.3.19 <Teacher> Add Question By Form 56](#_Toc531276062)

[2.3.20 <Teacher> Add Question By Import File 58](#_Toc531276063)

[2.3.21 <Teacher> Remove Questions 59](#_Toc531276064)

[2.3.22 <Teacher> Edit Questions 60](#_Toc531276065)

[2.3.23 <Teacher> Get List Test Exam 63](#_Toc531276066)

[2.3.24 <Teacher> Approve Test Exam Quality 64](#_Toc531276067)

[3. Software System Attribute 66](#_Toc531276068)

[3.1. Usability 66](#_Toc531276069)

[3.2. Reliability 66](#_Toc531276070)

[3.3. Availability 66](#_Toc531276071)

[3.4. Security 66](#_Toc531276072)

[3.5. Maintainability 66](#_Toc531276073)

[3.6. Portability 66](#_Toc531276074)

[3.7. Performance 66](#_Toc531276075)

[4. Conceptual Diagram 66](#_Toc531276076)

[D. Software Design Description 68](#_Toc531276077)

[1. Design Overview 68](#_Toc531276078)

[2. System Architecture Design 68](#_Toc531276079)

[2.1. Web Server Architecture Design 69](#_Toc531276080)

[2.2. Client Architecture Design 69](#_Toc531276081)

[3. Component Diagram 70](#_Toc531276082)

[4. Detailed Description 70](#_Toc531276083)

[4.1. Class Diagram 70](#_Toc531276084)

[4.2. Class Diagram Explanation 73](#_Toc531276085)

[4.2.1 Semester 73](#_Toc531276086)

[4.2.2 Exam 73](#_Toc531276087)

[4.2.3 ExamStudent 74](#_Toc531276088)

[4.2.4 Student 74](#_Toc531276089)

[4.2.5 BaseTest 74](#_Toc531276090)

[4.2.6 BaseTestContent 75](#_Toc531276091)

[4.2.7 Test 76](#_Toc531276092)

[4.2.8 TestContent 76](#_Toc531276093)

[4.2.9 Question 77](#_Toc531276094)

[4.2.10 Option 77](#_Toc531276095)

[4.2.11 Level 78](#_Toc531276096)

[4.2.12 QuestionType 78](#_Toc531276097)

[4.2.13 QuestionGroup 78](#_Toc531276098)

[4.2.14 QuestionObjective 79](#_Toc531276099)

[4.2.15 Chapter 79](#_Toc531276100)

[4.2.16 Course 79](#_Toc531276101)

[4.2.17 MainObjective 79](#_Toc531276102)

[4.2.18 LearningOutcome 80](#_Toc531276103)

[4.3. Interaction Diagram 80](#_Toc531276104)

[4.3.1 Activity Diagram 80](#_Toc531276105)

[4.3.2 Sequence Diagram 80](#_Toc531276106)

[5. Interface 80](#_Toc531276107)

[6. Database Design 82](#_Toc531276108)

[6.1. Entity Relationship Diagram 82](#_Toc531276109)

[6.2. Data Dictionary 83](#_Toc531276110)

[6.3. Class Diagram Explanation 84](#_Toc531276111)

[6.3.1 Semester 84](#_Toc531276112)

[6.3.2 Exam 84](#_Toc531276113)

[6.3.3 Student 84](#_Toc531276114)

[6.3.4 BaseTest 85](#_Toc531276115)

[6.3.5 BaseTestContent 85](#_Toc531276116)

[6.3.6 Test 86](#_Toc531276117)

[6.3.7 TestContent 86](#_Toc531276118)

[6.3.8 Question 86](#_Toc531276119)

[6.3.9 Option 87](#_Toc531276120)

[6.3.10 Level 87](#_Toc531276121)

[6.3.11 QuestionType 87](#_Toc531276122)

[6.3.12 QuestionGroup 88](#_Toc531276123)

[6.3.13 QuestionObjective 88](#_Toc531276124)

[6.3.14 Chapter 88](#_Toc531276125)

[6.3.15 Course 88](#_Toc531276126)

[6.3.16 MainObjective 89](#_Toc531276127)

[6.3.17 LearningOutcome 89](#_Toc531276128)

[7. Algorithms 90](#_Toc531276129)

[7.1. Calculating Percentage Similarity of 2 Question 90](#_Toc531276130)

[7.1.1 Problem 90](#_Toc531276131)

[7.1.2 Solution 90](#_Toc531276132)

[E. System Implementation & Test 91](#_Toc531276133)

[1. Introduction 91](#_Toc531276134)

[1.1. Overview 91](#_Toc531276135)

[1.2. Test Approach 91](#_Toc531276136)

[2. Database Relationship Diagram 91](#_Toc531276137)

[2.1. Physical Diagram: 91](#_Toc531276138)

[2.2. Data Dictionary: 93](#_Toc531276139)

[3. Performance Measures 93](#_Toc531276140)

[3.1. Parsing Question 93](#_Toc531276141)

[3.2. Validate Parsed Questions 93](#_Toc531276142)

[3.3. Matching Questions of a Course 93](#_Toc531276143)

[3.4. Random Question to Create Exam Base on Chapters/ Learning Outcome 93](#_Toc531276144)

[3.5. Get Exam Test from Server 93](#_Toc531276145)

[3.6. Submit Exam Tests of a Class 93](#_Toc531276146)

[4. Test Plan 93](#_Toc531276147)

[4.1. Features to be tested 93](#_Toc531276148)

[4.2. Features not to be tested 93](#_Toc531276149)

[5. System Testing Test Case 94](#_Toc531276150)

[5.1. Manage Exam System Test Case 94](#_Toc531276151)

[5.1.1 Import Question 94](#_Toc531276152)

[6. System Implement 100](#_Toc531276153)

[6.1. Web Server Application 100](#_Toc531276154)

[6.2. Web Client Application 101](#_Toc531276155)

[F. Software User’s Manual 103](#_Toc531276156)

[1. Installing Guide 103](#_Toc531276157)

[2. User Guide 106](#_Toc531276158)

[2.1. Admin Create Semester 106](#_Toc531276159)

[2.2. Teacher approve/reject exam 109](#_Toc531276160)

[2.3. Teacher Create Exam 112](#_Toc531276161)

[G. Appendix 116](#_Toc531276162)

List Of Tables

[Table 1: Definitions, Acronyms, and Abbreviations 12](#_Toc531728520)

[Table 2: Role and Responsibilities 15](#_Toc531728521)

[Table 3: Hardware Requirements 17](#_Toc531728522)

[Table 4 Software Requirement 17](#_Toc531728523)

[Table 5 Roles and Responsibilities Detail 19](#_Toc531728524)

[Table 6 Tools and Techniques 20](#_Toc531728525)

[Table 7 Product Backlog 20](#_Toc531728526)

[Table 8 Sprint 1 Development 21](#_Toc531728527)

[Table 9 SPRINT 2 DEVELOPMENT 22](#_Toc531728528)

[Table 10 SPRINT 3 DEVELOPMENT 23](#_Toc531728529)

[Table 11 SPRINT 4 DEVELOPMENT 24](#_Toc531728530)

[Table 12 SPRINT 5 DEVELOPMENT 25](#_Toc531728531)

[Table 13 SPRINT 6 DEVELOPMENT 25](#_Toc531728532)

[Table 14 <IT> Get List Process Must Kill 33](#_Toc531728533)

[Table 15 Add Must Be Killed Process to List 36](#_Toc531728534)

[Table 16 ADD TO LIST EXCEPTION PROCESS 37](#_Toc531728535)

[Table 17 Kill Process 39](#_Toc531728536)

[Table 18 <IT> GET REALTIME PROCESS 40](#_Toc531728537)

[Table 19 <IT> SEARCH REALTIME PROCESS 41](#_Toc531728538)

[Table 20 <IT> GET STUDENT SCREEN 43](#_Toc531728539)

[Table 21 <IT> Submit Student Test Exam 44](#_Toc531728540)

[Table 23 <IT> SEARCH ONLINE STUDENT 46](#_Toc531728541)

[Table 24 <TRAINING DEPARTMENT STAFF> ADD NEW SEMESTER 47](#_Toc531728542)

[Table 25 <Training Department Staff> Get Semester Detail 49](#_Toc531728543)

[Table 26 <MANAGER> APPROVE AVAILABLE TEST EXAM 51](#_Toc531728544)

[Table 27 <Leader> Search Chapter 53](#_Toc531728545)

[Table 28 Add chapter 55](#_Toc531728546)

[Table 29 Delete chapter 56](#_Toc531728547)

[Table 30 <Leader> Edit chapter 59](#_Toc531728548)

[Table 31 <Teacher> Search Question 61](#_Toc531728549)

[Table 32 <TEACHER> ADD QUESTION BY FORM 63](#_Toc531728550)

[Table 33 <Teacher> Add Question by Import File 65](#_Toc531728551)

[Table 34 <Teacher> Remove Questions 66](#_Toc531728552)

[Table 35 <Teacher> Edit Questions 68](#_Toc531728553)

[Table 36 <Teacher> Get List Test Exam 70](#_Toc531728554)

[Table 37 <Teacher> Approve Test Exam 73](#_Toc531728555)

[Table 38 <Training department staff> Generate Test Exam 75](#_Toc531728556)

[Table 38 <Training department staff> Assign Approver 76](#_Toc531728557)

[Table 39 CONCEPTUAL DIAGRAM EXPLAINATION 79](#_Toc531728558)

[Table 40 ARCHITECTURE OVERVIEW DESCRIPTION 81](#_Toc531728559)

[Table 41 Class diagram Description 85](#_Toc531728560)

[Table 42 Class Semester Explanation 85](#_Toc531728561)

[Table 43 Class Exam Explanation 86](#_Toc531728562)

[Table 44 Class ExamStudent Explanation 86](#_Toc531728563)

[Table 45 CLASS STUDENT EXPLANATION 86](#_Toc531728564)

[Table 46 CLASS BASE TEST EXPLANATION 87](#_Toc531728565)

[Table 47 CLASS BASETESTCONTENT EXPLANATION 88](#_Toc531728566)

[Table 48 CLASS TEST EXPLANATION 88](#_Toc531728567)

[Table 49 CLASS TESTCONTENT EXPLANATION 89](#_Toc531728568)

[Table 50 CLASS QUESTION EXPLANATION 89](#_Toc531728569)

[Table 51 CLASS OPTION EXPLANATION 90](#_Toc531728570)

[Table 52 CLASS OPTION EXPLANATION 90](#_Toc531728571)

[Table 53 CLASS QUESTIONTYPE EXPLANATION 90](#_Toc531728572)

[Table 54 CLASS QUESTION GROUP EXPLANATION 91](#_Toc531728573)

[Table 55 CLASS QUESTIONOBJECTIVE EXPLANATION 91](#_Toc531728574)

[Table 56 CLASS CHAPTER EXPLANATION 91](#_Toc531728575)

[Table 57 CLASS COURSE EXPLANATION 91](#_Toc531728576)

[Table 58 CLASS MAINOBJECTIVE EXPLANATION 92](#_Toc531728577)

[Table 59 CLASS LEARNINGOUTCOME EXPLANATION 92](#_Toc531728578)

[Table 60 CLASS TeacherTest EXPLANATION 93](#_Toc531728579)

[Table 61 CLASS TeacherTestContent EXPLANATION 93](#_Toc531728580)

[Table 60 ERD Dictionary 97](#_Toc531728581)

[Table 61 Semester table Description 97](#_Toc531728582)

[Table 62 Exam table Description 97](#_Toc531728583)

[Table 63 Student table Description 98](#_Toc531728584)

[Table 64 BaseTest table Description 98](#_Toc531728585)

[Table 65 BaseTestContent table Description 99](#_Toc531728586)

[Table 66 Test table Description 99](#_Toc531728587)

[Table 67 TestContent table Description 100](#_Toc531728588)

[Table 68 Question table Description 100](#_Toc531728589)

[Table 69 Option table Description 101](#_Toc531728590)

[Table 70 Level table Description 101](#_Toc531728591)

[Table 71 QuestionType table Description 101](#_Toc531728592)

[Table 72 QuestionGroup table Description 101](#_Toc531728593)

[Table 73 QuestionObjective table Description 102](#_Toc531728594)

[Table 74 Chapter table Description 102](#_Toc531728595)

[Table 75 Course table Description 102](#_Toc531728596)

[Table 76 MainObjective table Description 102](#_Toc531728597)

[Table 77 LearningOutcome table Description 103](#_Toc531728598)

[Table 80 LearningOutcome table Description 103](#_Toc531728599)

[Table 81 TeacherTest table Description 104](#_Toc531728600)

[Table 82 TeacherTestContent table Description 104](#_Toc531728601)

[Table 78 Physical Diagram Description 112](#_Toc531728602)

[Table 79 SEMESTER TABLE DESCRIPTION 112](#_Toc531728603)

[Table 80 Exam Entity Description 112](#_Toc531728604)

[Table 81 Student Entity Description 113](#_Toc531728605)

[Table 82 BaseTest Entity Description 113](#_Toc531728606)

[Table 83 BaseTestContent Entity Description 114](#_Toc531728607)

[Table 84 Test Entity Description 114](#_Toc531728608)

[Table 85 TestContent Entity Description 114](#_Toc531728609)

[Table 86 Question Entity Description 115](#_Toc531728610)

[Table 87 Option Entity Description 115](#_Toc531728611)

[Table 88 Level Entity Description 116](#_Toc531728612)

[Table 89 QuestionType Entity Description 116](#_Toc531728613)

[Table 90 QuestionGroup Entity Description 116](#_Toc531728614)

[Table 91 QuestionObjective Entity Description 116](#_Toc531728615)

[Table 92 Chapter Entity Description 117](#_Toc531728616)

[Table 93 Course Entity Description 117](#_Toc531728617)

[Table 94 MainObjective Entity Description 117](#_Toc531728618)

[Table 95 LearningOutcome Entity Description 117](#_Toc531728619)

[Table 96 AuthenticationApp table Description 118](#_Toc531728620)

[Table 97 LearningOutcome Entity Description 118](#_Toc531728621)

[Table 98 LearningOutcome Entity Description 118](#_Toc531728622)

[Table 99 Import Question Test Result 1 121](#_Toc531728623)

[Table 100 Import question test result 2 121](#_Toc531728624)

[Table 101 Import question test result 3 121](#_Toc531728625)

[Table 102 Matching Question test result 122](#_Toc531728626)

[Table 103 Generate Test Result 122](#_Toc531728627)

[Table 104 Approve Exam Test result 123](#_Toc531728628)

[Table 105 Run exam tool test result 1 124](#_Toc531728629)

[Table 106 Run exam tool test result 2 124](#_Toc531728630)

[Table 107 Run exam tool test result 3 124](#_Toc531728631)

[Table 108 Take Exam test result 1 125](#_Toc531728632)

[Table 109 Take Exam test result 2 125](#_Toc531728633)

[Table 110 Take Exam test result 3 126](#_Toc531728634)

List of Figures

[Figure 1 The Scrum Process 19](#_Toc531728437)

[Figure 2 Use case overview Examination System 27](#_Toc531728438)

[Figure 3 Teacher use case Detail 28](#_Toc531728439)

[Figure 4 Leader use case detail 29](#_Toc531728440)

[Figure 5 IT use case detail 29](#_Toc531728441)

[Figure 6 Manager use case detail 30](#_Toc531728442)

[Figure 7 Student use case detail 30](#_Toc531728443)

[Figure 8 Training Department Staff use case detail 31](#_Toc531728444)

[Figure 9 <IT> Get List Process Must Kill 32](#_Toc531728445)

[Figure 10 Add Must Be Killed Process to List 34](#_Toc531728446)

[Figure 11 Add to List Exception Process 36](#_Toc531728447)

[Figure 12 Kill Process 38](#_Toc531728448)

[Figure 13 <IT> Get Realtime Process 39](#_Toc531728449)

[Figure 14 <IT> Search Realtime Process 40](#_Toc531728450)

[Figure 15 <IT> Get Student Screen 41](#_Toc531728451)

[Figure 16 <IT> Submit Student Test Exam 43](#_Toc531728452)

[Figure 17 <IT> Search Online Student 44](#_Toc531728453)

[Figure 18 <Training Department Staff> Add new Semester 46](#_Toc531728454)

[Figure 19 <Training Department Staff> Get Semester Detail 48](#_Toc531728455)

[Figure 20 <Manager> Public Test Exam 49](#_Toc531728456)

[Figure 21 <Leader> Search Chapter 51](#_Toc531728457)

[Figure 22 Add chapter 53](#_Toc531728458)

[Figure 23 Delete chapter 55](#_Toc531728459)

[Figure 24 <Leader> Edit chapter 57](#_Toc531728460)

[Figure 25 <Teacher> Search Question 59](#_Toc531728461)

[Figure 26 <Teacher> Add Question by Form 61](#_Toc531728462)

[Figure 27 <Teacher> Add Question by Import File 63](#_Toc531728463)

[Figure 28 <Teacher> Remove Questions 65](#_Toc531728464)

[Figure 29 <Teacher> Edit Questions 67](#_Toc531728465)

[Figure 30 <Teacher> Get List Test Exam 69](#_Toc531728466)

[Figure 31 <Teacher> Approve Test Exam 71](#_Toc531728467)

[Figure 31 <Teacher> Approve Test Exam 73](#_Toc531728468)

[Figure 31 < Training Department Staff > Approve Test Exam 75](#_Toc531728469)

[Figure 32 Conceptual Diagram 78](#_Toc531728470)

[Figure 33 System Architecture Design 80](#_Toc531728471)

[Figure 34 Architecture Overview 81](#_Toc531728472)

[Figure 35 Web Admin MVC Architecture 82](#_Toc531728473)

[Figure 36 Class diagram 83](file:////Volumes/Document/Capston/document/ExamTool.docx#_Toc531728474)

[Figure 37 ERD DIAGRAM 95](#_Toc531728475)

[Figure 38 Levenshtein distance example result 108](#_Toc531728476)

[Figure 39 Onion Architecture 127](#_Toc531728477)

[Figure 40 Onion Architecture description 127](#_Toc531728478)

[Figure 41 Angular Component Example 128](#_Toc531728479)

Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| L.O | Learning Outcome |
| MVC | Model View Controller |
| API | Application Programming Interface |
| IDE | Integrated development environment |
| GUI | Graphic user interface |
| ERD | Entity relationship diagram |
| I/O | Input/Output |

Table 1: Definitions, Acronyms, and Abbreviations

# Introduction

## Project Information

* Project name: **Examination Tools applying Block Chain Technology**
* Project Code: **ExamTool**
* Product Type: **Web Application, Desktop Application**
* Start Date: **September 11, 2018**
* End Date: **December 13, 2018**

## Introduction

In this document, we will introduce a solution for FPT University Examination system. The current system have some problems like it does not guarantee the quality of the result and it may be used by students to exploit vulnerabilities for cheating. Based on our researches and analysis, we proposed a new solution for FPT University Examination system.

We will organize questions, exams by Learning Outcome and apply block-chain technology in there. Learning Outcome are statements that describe the knowledge or skills which the students should acquire by the end of a particular assignment, class, course, or program and help students understand why those knowledge and skills will be useful to them. A test exam created with Learning Outcome will ensure the coverage of a course content inside it. Beside the insurance of the test exam content, we also ensure the test exam and the results will be encrypted by Block-chain technology. Block-chain is a technology which allows data to be transmitted safely and correctly by a very complex encryption. Block-chain will be used in the test exam approving phase to ensure that test exam will not be editable.

## Current Situation

Two weeks before the exam start, staff will create test for each subject. The current question management system is manage questions by chapters. Depending on the teacher's requirements, the staff will randomly select questions based on the topic or chapter. Such random selection like that can not guarantee the test for being covered with the entire knowledge of the subject. After generated, lecture of the subject will be notified and review the topic. During review process, lectures can remove or edit any questions in the test but the system could not update those modifications automatically in the question bank. Therefore, the reviewer has to edit the questions once over again in next time. If nothing is wrong, the test will be kept confidential until the exam. The reviewer has to sign in a document to confirm that the test can be used for the upcoming exam.

Upon exam arrival, students will enter the examination room, open the current examination software to take the exam. Before entering the room, students can turn on fraudulent applications and hide it. The examiners cannot know what students are doing outside the room so if the software is not in the blacklist it will not be destroyed by the exam software and students can use actions such as using shortcut to trigger an event to capture a test page, or to black out a self-contained search text message, or, more importantly, to use TeamViewer to complete the test. Such behaviors will help students pass the exam easily and not evaluate the real ability of students. After the exam finish, the exam software can add those cheating applications to the blacklist, students will find another application and so on, cannot prevent students from cheating during the course.

## Problem Definition

### Proposed Solution

* The topic is divided by topic and chapter, not cover all the content of a course.
* The question updating is repeated and not synchronized.
* Cannot prevent student from cheating during examinations

### Feature Functions

* **Random question base on L.O:** Lecture will select L.O first, then all questions will be random base on them.
* **Update question in question bank:** After leader edit questions in the exam, that questions in question bank will be updated.
* **Anti-cheating:** We will prevent unnecessary actions and all processes in the list which was inputted, will be killed.

## Values and Challenges Value

**Values**

* Provide better question bank management
* Generate exam that can cover all content of the course
* Kill processes which is not allowed while doing an exam.

**Challenges**

* Many import types is supported.
* Reduce cheating on exam.

## Functional Requirement

Functional requirements of the system are listed as below:

**Training Department Staff**

* Manage Exam
* Support generate examination test from the question banks.

**Teacher**

* Import question from some different type.
* Edit question and warning when duplicate.
* Approve test exam quality.

**Leader**

* Manage Course
* Manage Chapter
* Manage Learning Outcome

**Student**

* Support doing an exam
* Get result of exam when submit exam to server has problem

**IT**

* IT can manage student process

**System Handler**

* Reduced cheating
* Explicit generate test exam, exam, approve process by applying block chain.

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
|  |  |  |  |  |
| 1 | Kiều Trọng Khánh | Project Owner | Supervisor | khanhkt@fe.edu.vn |
|  |  |  |  |  |
| 2 | Nguyễn Minh Hưng | Scrum team members | Leader | hungnmse61794@fpt.edu.vn |
|  |  |  |  |  |
| 3 | Cao Trung Hiếu | Scrum master | Member | hieuctse61801@fpt.edu.vn |
|  |  |  |  |  |
| 4 | Trương Tấn Sang | Scrum team members | Member | sangttse61926@fpt.edu.vn |
|  |  |  |  |  |

Table 2: Role and Responsibilities

# Software Project Management Plan

## 1. Problem Definition

### Name of this Capstone Project

• **Official name**: Examination Tools applying Block Chain Technology.

• **Vietnamese name**: Xây dựng hệ thống quản lý đề thi và việc thi cử sử dụng công nghệ Block-Chain.

• **Abbreviation**: ExamTool.

### Problem Abstract

This project is created to assist the school to fight against cheating in the examination as well as develop an exam administration to be more optimal. In the process of research and analysis, we decided to implement ExamTool by using Extension and applying block chain in Test Exam result. But this way have a lot of problem because Extension can not intervene the system deeply and Blockchain is not suitable for test exam result. So that, we have to change the idea to using an application to kill processes of the system and applying blockchain in generate, approve test exam.

The Desktop application, which is built on Window, can deep embedding the system and kill unauthorized processes. Firstly, application will send a request to server to verify if this application is valid to access the website. After verified, application will be allowed to access resources for exam questions. Students will do the test during the given time.

Block-chain technology will be applied in from the generate test exam to approved test exam process, which make sure them will not to be lost or changed. For each bad behavior once being discovered during test time, we are going to deduct time or stop the test immediately.

Regarding to the question management function, we find it difficult to input the question to the question bank as well as organize the examination. Therefore, a variety of template formats are going to be provided in the word file, which facilitates the lecturers to import the file. Parsing process will be resolved by the clients. In addition, we also look up the potential duplicate questions before importing them into the question bank, then notify to the lecturer for resolve.

The tests will be generated based on the Learning Outcome or Chapter of a subject or manually selected by the staff. Thus, the exam will be guaranteed to contain all knowledge of a subject, then notify to the leaders for the approval. If any errors occur during the review process, the reviewer can directly edit the question right on the test. The question which has been corrected will be synchronized versus the question in the question bank.

### Project Overview

#### Current Situation

*Below are the problems encountered in this project:*

* **New techniques:** The team is new to block-chain techniques.
* **Lack of knowledge** about the window system.
* **Lack of knowledge** about examination system.

#### Boundaries of the system

*Our system supports:*

* **Import question** file with many question types.
* **Matching questions** and **show all questions** can be duplicated.
* **Synchronize question** in question bank after edited in exam.
* **Random questions** in exam **base on L.O or Chapter**.
* **Killing process** real-time.
* ExamTool run best for Window 10.

*Our system hasn’t supported:*

* ExamTool provide for others O.S like Linux or Mac
* Exam Management System do not support Internet Explorer and Microsoft Edge.

#### Future Plans

* Flexible to custom template import question.
* More feature to manage student take exam process.

#### Development Environment

##### Hardware requirements

|  |  |  |
| --- | --- | --- |
| Hardware | Minimum Requirements | Recommended |
| Internet Connection | Cable | Cable |
| Operating System | Window Server 2008 | Window Server 2016 |
| Computer Processor | Intel® Xeon ® 3.0GHz | Intel® Xeon ® Processors |
| Computer Memory | 4GB RAM | 8GB RAM or more |

Table 3: Hardware Requirements

##### Software requirements

|  |  |  |
| --- | --- | --- |
| Software | Name / Version | Description |
| Environment |  |  |
| Modeling tool | Star UML |  |
| IDE | Visual Studio Enterprise 2017  Visual Studio Code 1.23.1  Webstorm 2018.2 | Programming tools |
| DBMS | SQL Server 2017 | Used to create & manage the database for system |
| Source control | Tortoise SVN 1.10.1 | Used for source control |

Table 4 Software Requirement

## Coding Convention

**JavaScript:**

*Naming Conventions:*

* Variable and function names written as **camelCase**.
* Global variables written in **UPPERCASE**.
* Constants written in **UPPERCASE**.

*Function Conventions:*

* Put the opening bracket at the end of the first line.
* Use one space before the opening bracket.
* Put the closing bracket on a new line, without leading spaces.
* Do not end a complex statement with a semicolon.

*Others***:**

* Always put spaces around operators ( = + - \* / ), and after commas.
* Always use 4 spaces for indentation of code blocks.
* Always end a simple statement with a semicolon.
* Always end an object definition with a semicolon.

**C#:**

*Naming Conventions:*

* Use **PascalCase** for public property, method and type name.
* For parameters and local variables, use **camelCase**.
* For private fields, use prefix \_**camelCase** with an **\_**.
* Named constants with **ALLCAPS**.
* Vertically align curly brackets.
* Prefix interfaces with the letter **I**. Interface names are noun (phrases) or adjectives.

**Reference:**

* [C# Style Guide and Coding Convention](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions)
* [JavaScripts Style Guide and Coding Convention](https://www.w3schools.com/js/js_conventions.asp)

## Project Organization

### Software Process Model

This project is developed using the Scrum model – part of an agile framework for Software development project. Our team chooses the Scrum model because of the following reasons:

* Our team only has 3 members, and tasks are assigned vertically, do all steps from design, coding, testing, and implementation. Scrum is the most suitable model for the small and medium project.
* In the project, there are many new technologies that need to be learned. With the Scrum model, the team can learn and develop in parallel to meet the deadline.
* The product owner can change the requirement or extend scope. The team will adapt to change better.

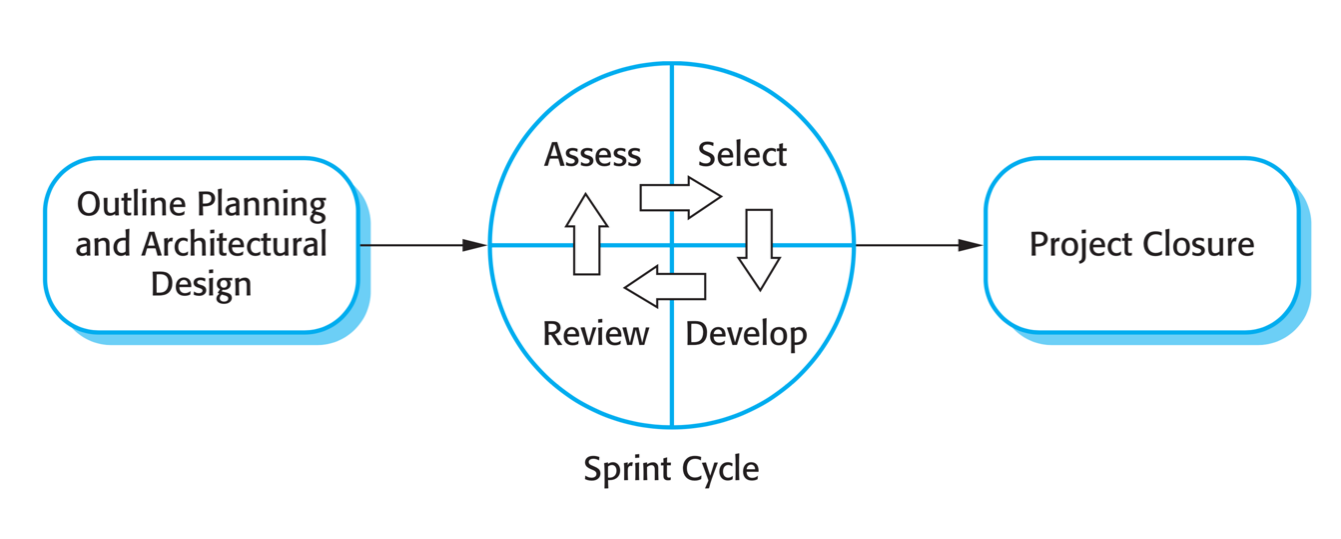


Figure 1 The Scrum Process

*Reference: Software Engineering 9th by Somerville, page 73*

### Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in group** | **Responsibilities** |
| 1 | Kiều Trọng Khánh | Project Owner | * Specify scope and user requirement * Give out technique and business analysis support * Control the development process |
| 2 | Cao Trung Hiếu | Scrum master | * Create Sprint Backlog and Product Backlog * Make sure the Scrum teams understand and follow the process. * Help the team master scrum artifacts such as Sprint Backlog, Product Backlog, ... * Writing report * Always be present to answer questions and give advice when product owner or scrum member needs. |
| 3 | Nguyễn Minh Hưng  Trương Tấn Sang | Scrum team members | * Clarifying requirements * Prepare documents * Designing database * GUI Design * Coding * Testing |

Table 5 Roles and Responsibilities Detail

### Tools and Techniques

|  |  |
| --- | --- |
| Tool/Technique | Name and version |
| Front-end | Angular 2+, Jquery, boostrap |
| Back-end | .Net framework 4.7.2 |
| IDE | Visual Studio Enterprise 2017, Visual Studio Code 1.23.1, Webstorm 2018.2 |
| Database | SQL Server 2017 |
| Modeling Tool | Star UML |
| Version Control | Tortoise SVN 1.10.1 |
| Task Management | Trello |

Table 6 Tools and Techniques

## Project Management Plan

### Product Backlog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprint** | **Story ID** | **Story** | **Task ID** | **Task** |
| 1 | 1 | Introduction document | 1.1 | Project Information |
|  |  |  | 1.2 | Introduction |
|  |  |  | 1.3 | Current Situation |
|  |  |  | 1.4 | Problem Definition |
|  |  |  | 1.5 | Proposed Solution |
|  |  |  | 1.6 | Role and Responsibility |
|  |  |  | 1.7 | Functional Requirements |
|  | 2 | Product Backlog | 2.1 | Create Product Backlog |
|  | 3 | Project management plan | 3.1 | Problem Definition |
|  |  |  | 3.2 | Project Organization |
|  |  |  | 3.3 | Project management plan |
|  |  |  | 3.4 | Coding Convention |

Table 7 Product Backlog

### Sprint Backlog

#### Sprint 1 (10.09.2018 – 23.09.2018): Project initiation

##### Goal

1.1 Project Information

1.2 Introduction

1.3 Current Situation

1.4 Problem Definition

1.5 Proposed Solution

1.6 Role and Responsibility

1.7 Functional Requirements

2.1 Create Product Backlog

3.1 Problem Definition

3.2 Project Organization

3.3 Project management plan

3.4 Coding Convention

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | Project Information | HungNM |
| 1.2 | Introduction | HungNM |
| 1.3 | Current Situation | HungNM |
| 1.4 | Problem Definition | HieuCT |
| 1.5 | Proposed Solution | HieuCT |
| 1.6 | Role and Responsibility | SangTT |
| 1.7 | Functional Requirements | SangTT |
| 2.1 | Create Product Backlog | HieuCT |
| 3.1 | Problem Definition | HungNM |
| 3.2 | Project Organization | HieuCT |
| 3.3 | Project management plan | HieuCT, SangTT |
| 3.4 | Coding Convention | SangTT |

Table 8 Sprint 1 Development

#### Sprint 2 (24.09.2018 – 8.10.2018): Software Document

##### Goal

1.1 User Requirement Specification

1.2 Software Requirement Specification

1.3 Software System attribute

1.4 Conceptual diagram

1.5 Design Overview

1.6 System Architectural Design

1.7 Component Diagram

2.1 Detailed Description

3.1 Interface

3.2 Interface Design

3.3 Algorithms

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | User Requirement Specification | HungNM |
| 1.2 | Software Requirement Specification | HungNM |
| 1.3 | Software System attribute | HungNM |
| 1.4 | Conceptual diagram | HieuCT |
| 1.5 | Design Overview | HieuCT |
| 1.6 | System Architectural Design | SangTT |
| 1.7 | Component Diagram | SangTT |
| 2.1 | Detailed Description | HieuCT |
| 3.1 | Interface | HungNM |
| 3.2 | Interface Design | HieuCT, HungNM |
| 3.3 | Algorithms | HieuCT, SangTT |

Table 9 SPRINT 2 DEVELOPMENT

#### Sprint 3 (9.10.2018 – 23.10.2018): Basic feature and core

##### Goal

1.1 System Problem definition

1.2 Implement Algorithms

1.3 Import question from Moodle

1.4 Manage Exam

1.5 Manage Course

1.6 Manage Question

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | System Problem definition | HieuCT |
| 1.2 | Implement Algorithms | HieuCT |
| 1.3 | Import question from Moodle | HungNM |
| 1.4 | Manage Exam | HungNM, SangTT |
| 1.5 | Manage Question | HungNM, SangTT |

Table 10 SPRINT 3 DEVELOPMENT

#### Sprint 4 (24.10.2018 – 8.11.2018): Software Document

##### Goal

1.1 Teacher Import and Manage Question

1.2 Manage LO

1.3 Manage Chapter

1.4 Staff generate test exam

1.5 Student App Implement

1.6 Realtime kill Process Implement

1.7 Leader Approve Test Exam

2.1 Student Take Exam

3.1 Review Student Test Exam

3.2 Blockchain Implement

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | Teacher Import and Manage Question | HungNM |
| 1.2 | Manage LO | SangTT, HungNM |
| 1.3 | Manage Chapter | SangTT, HungNM |
| 1.4 | Staff generate test exam | SangTT, HungNM |
| 1.5 | Student App Implement | HieuCT |
| 1.6 | Realtime kill Process Implement | HieuCT |
| 1.7 | Leader Approve Test Exam | SangTT, HungNM |
| 2.1 | Student Take Exam | HieuCT |
| 3.1 | Review Student Test Exam | HieuCT |
| 3.2 | Blockchain Implement | SangTT |

Table 11 SPRINT 4 DEVELOPMENT

#### Sprint 5 (9.11.2018 – 23.11.2018): Software Document

##### Goal

1.1 IT Get Student Screen

1.2 Login with FPT API

1.3 Manager Approve Exam

1.4 Run test With FPT Student

1.5 Report bug

1.6 Fix bug

1.7 Test document

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | IT Get Student Screen | HieuCT |
| 1.2 | Login with FPT API | SangTT, HungNM |
| 1.3 | Manager Approve Exam | SangTT, HungNM |
| 1.4 | Run test With FPT Student | SangTT, HungNM |
| 1.5 | Report bug | HieuCT |
| 1.6 | Fix bug | HieuCT |
| 1.7 | Test document | SangTT, HungNM |

Table 12 SPRINT 5 DEVELOPMENT

#### Sprint 6 (24.11.2018 – 13.12.2018): Complete Document

##### Goal

1.1 Installation Guide

1.2 User Manual

1.3 Paper Document

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 1.1 | Installation Guide | HieuCT |
| 1.2 | User Manual | SangTT, HungNM |
| 1.3 | Paper Document | SangTT, HungNM |

Table 13 SPRINT 6 DEVELOPMENT

# Software Requirement Specification

## User Requirement Specification

### Examination Requirement

* + - * 1. The examination system can do the following functions:

Allow students taking and submitting exams.

Allow IT support summit student Exam when have technical problem.

Block virtual machines.

Block special key strokes.

Ensure students exam doing process by saving it locally with advance hashing algorithm which password is needed to decrypt.

Reconnect to FPT University wifi automatically when disconnected during exam.

### System Requirement

* + - * 1. The system does the following functions:

Categorize questions by Learning Outcome.

Secure and ensure exam creation process step by step by using Blockchain technology.

Kill process in the process must kill list when students start the ExamTool.

Detect similar questions before import it to the question bank and notify to importer.

Synchronize questions between question bank and exam bank.

### IT Department Requirement

* + - * 1. The administrator does the following functions:

Manage student’s computer current process.

View student’s screen while they are taking exams.

View connection process of students which are taking exams.

Add 3rd party processes to kill list.

Submit a student exam which can also be done by IT.

## System Requirement Specification

### External Interface Requirement

#### User Interface

* + - * 1. The user interface uses language is English for all web application and windows application.

#### Hardware Interface

* + - * 1. N/A

#### Software Interface

* + - * 1. Service 3rd party:

Microsoft SignalR 2

FPT Authenticate

#### Communication Protocol

* + - * 1. Use HTTP protocol 1.1 for communication between:

Web Application and Web Server.

Windows Application and Web Server.

### System Overview Diagram

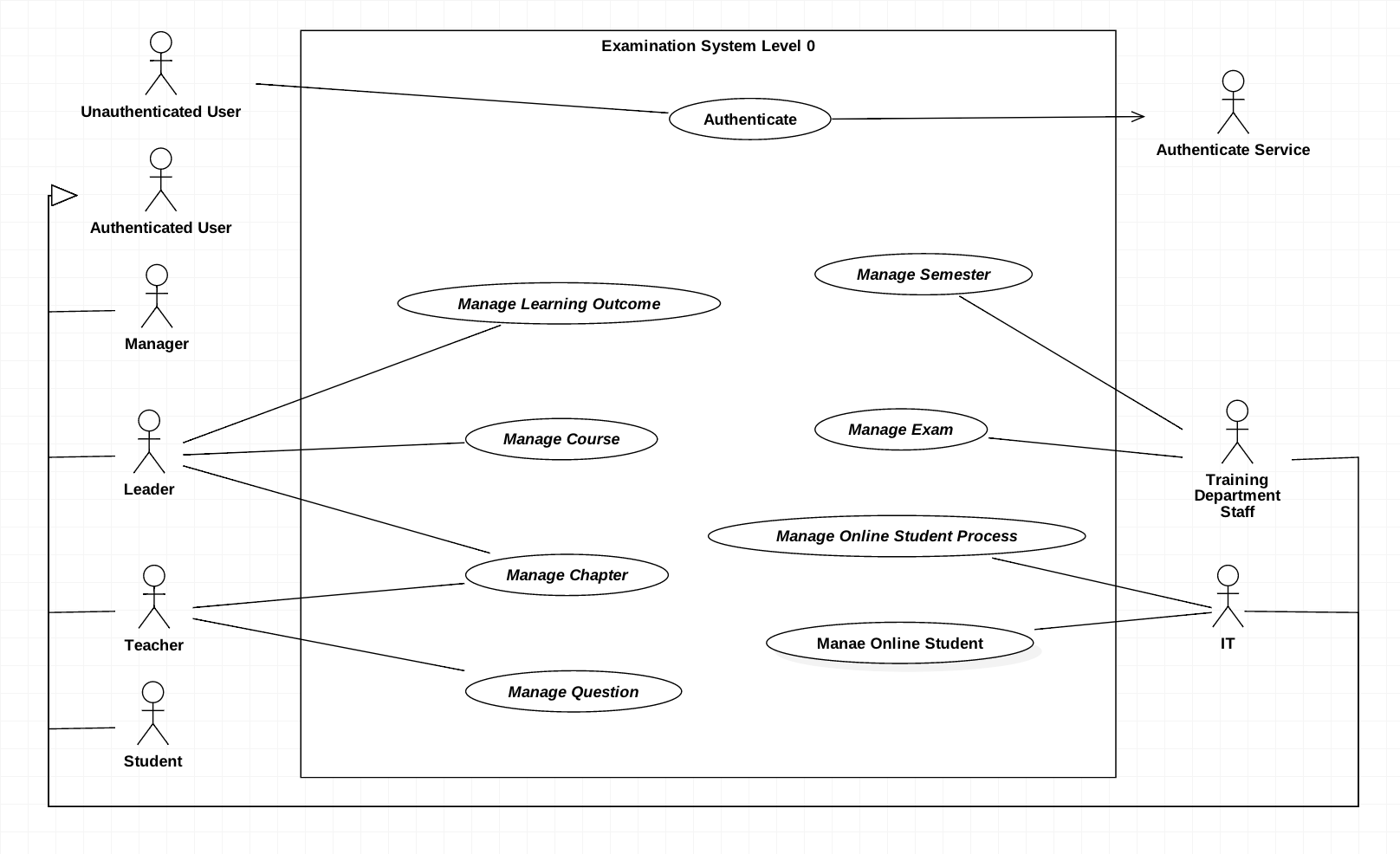
****

Figure 2 Use case overview Examination System

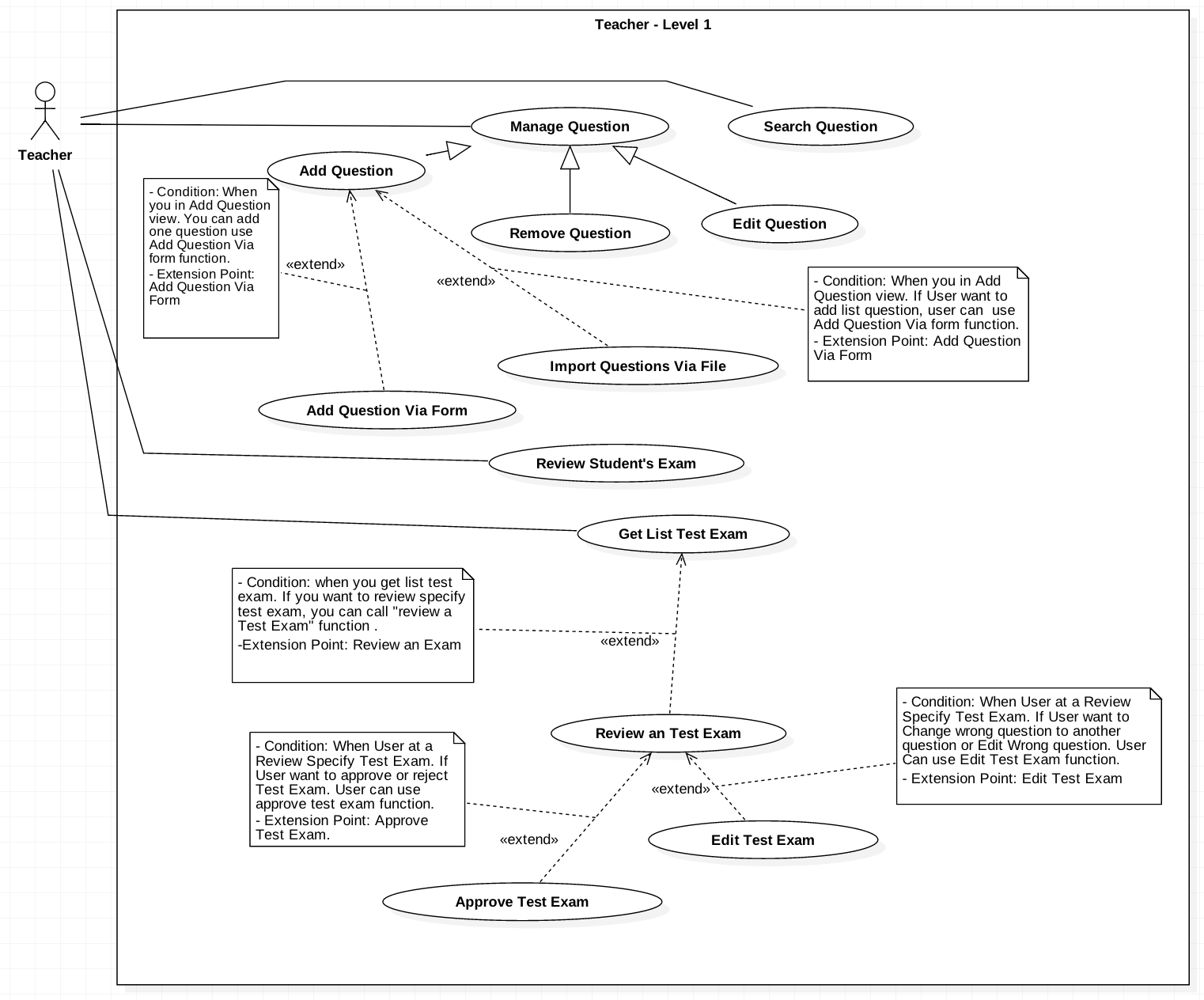


Figure 3 Teacher use case Detail

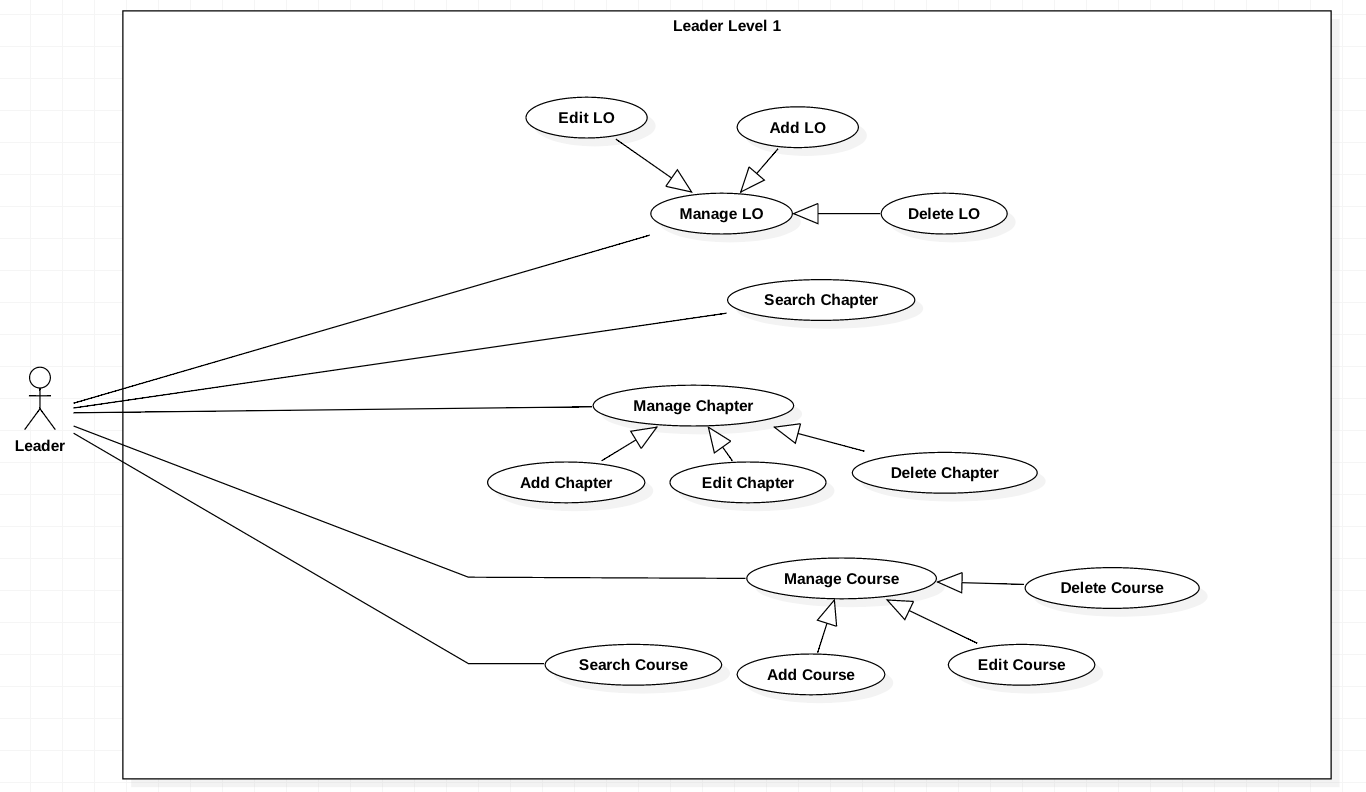
******

Figure 4 Leader use case detail

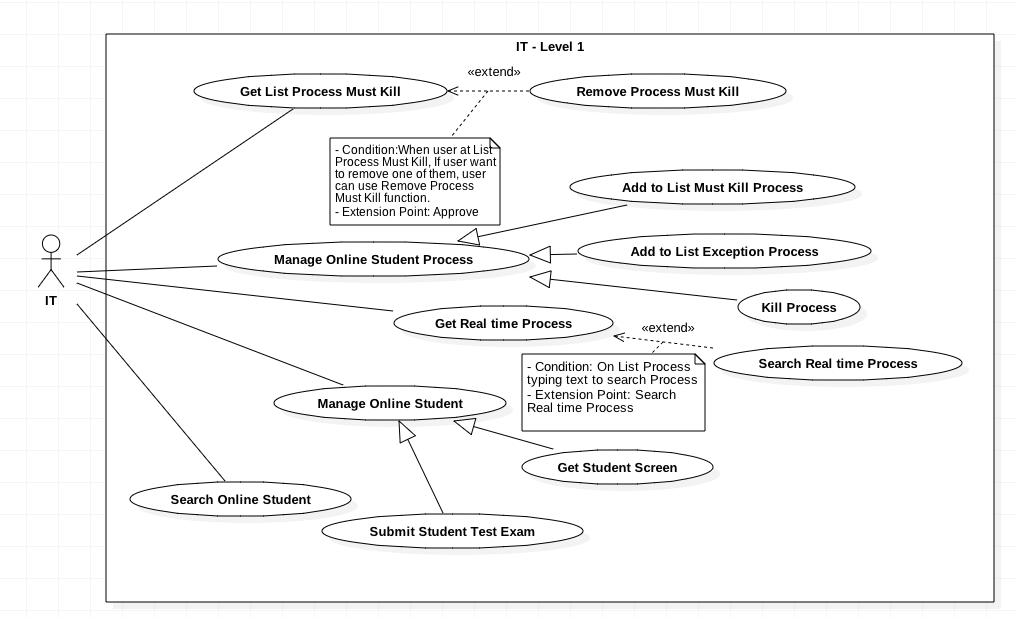
******

Figure 5 IT use case detail

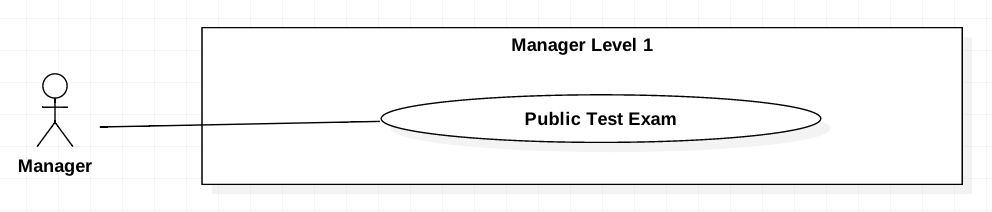


Figure 6 Manager use case detail

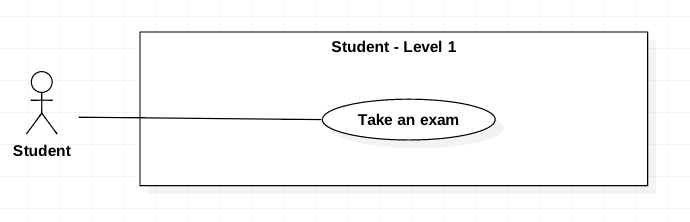
******

Figure 7 Student use case detail

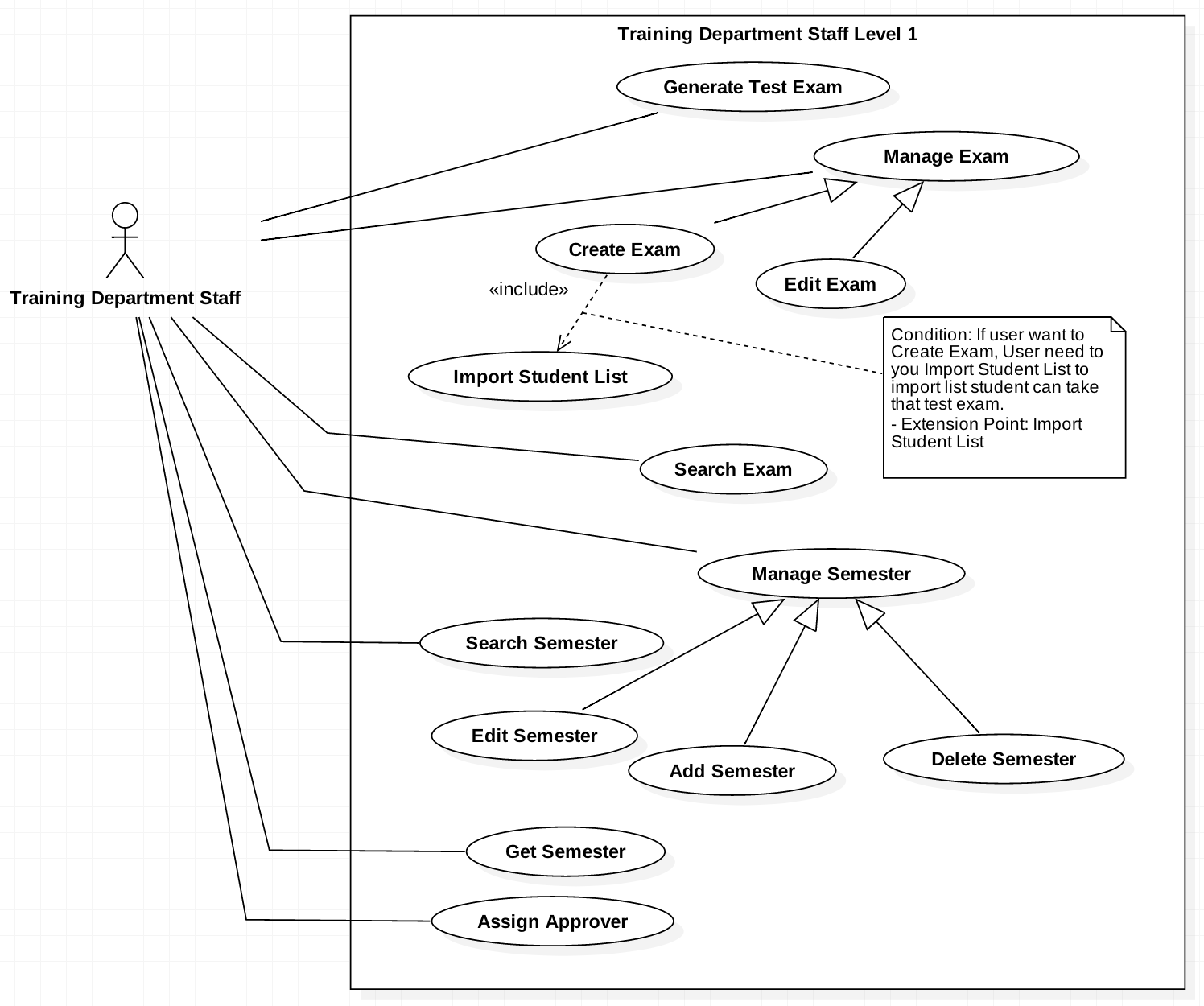


Figure 8 Training Department Staff use case detail

### List of use case

#### <IT> Get List Process Must Kill

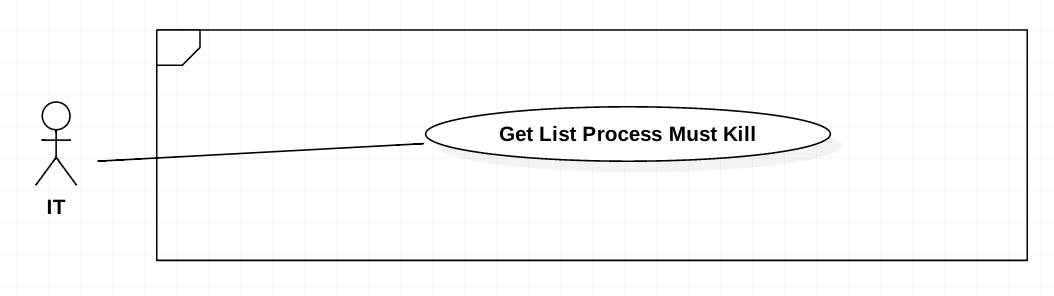
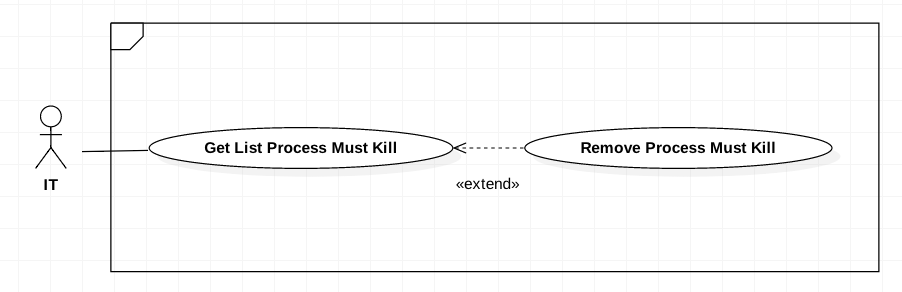


Figure 9 <IT> Get List Process Must Kill

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.01** | | | |
| **Use Case No.** | UC\_IT01 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get List Processes Must Kill | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * IT   **Summary:**   * *This use case allows IT Staff to Get list process must kill.* * *“Process must kill” is processes will be killed when the student run the ExamTool.*   **Goal:**   * *IT can check all of processes which must be killed they recorded for preventing cheating.*   **Triggers:**   * *IT call command to server.*   **Preconditions:**   * *IT must record processes which they defined cheating processes.*   **Post Conditions:**   * *Success: Get list of processes must be killed.* * *Fail: Show error message*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to the tab “process must kill screen”* |  | | *2* |  | *The system will be displayed list process must kill with information:*   * ProcessName * ApplicationName * ProcessPath |   **Alternative Scenario:** *N/A*  **Relationship:** *N/A*  **Business Rules:**   * *IT Managers can get a list of processes must be killed in the database.* | | | |

Table 14 <IT> Get List Process Must Kill

#### <IT> Remove Process Must be Killed



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.02** | | | |
| **Use Case No.** | UC\_IT02 | **Use Case Version** | 2.0 |
| **Use Case Name** | Remove Process Must be Killed | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT to remove some process from process must killed list.*   **Goal:**   * *IT can remove process in list which must be killed they recorded for preventing cheating.*   **Triggers:**   * *IT call command to server.*   **Preconditions:**   * *IT must record processes which they defined cheating processes*.   **Post Conditions:**   * *Success: Process must be removed from list.* * *Fail: Show error message*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to the tab “process must kill screen”* |  | | *2* |  | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* | | *3* | *Select the process row which IT want to remove* |  | | *4* | *IT send remove from process must kill command* | *The system will remove selected row from database and display new list in interface.* |   **Relationships:**   * *Get List Processes Must Kill*   **Business Rules:**   * *IT Managers can remove any process in the process kill list if they see it’s no longer being harmful.* * *Process must be inside the process kill list.* | | | |

#### <IT> Add Must Be Killed Process to List

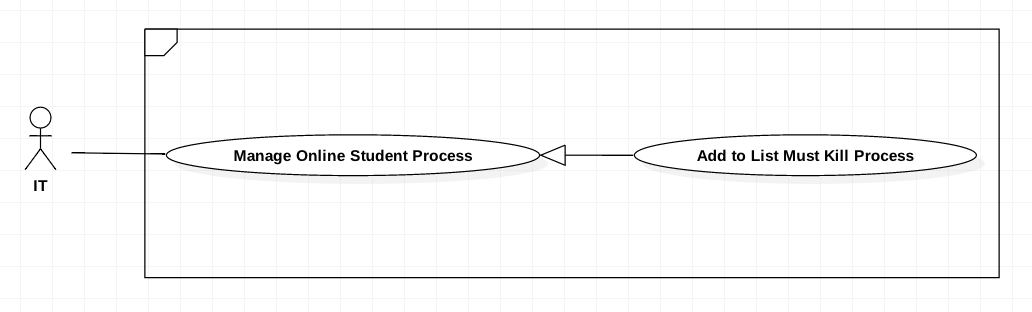


Figure 10 Add Must Be Killed Process to List

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.03** | | | |
| **Use Case No.** | UC\_IT03 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add to List Must be Killed Process | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT can add new process to process must killed list.*   **Goal:**   * *Add a new process to list process must be killed.*   **Trigger:**   * *IT sends create command*   **Preconditions:**   * *IT scan student computers ‘s running processes.*   **Post Conditions:**   * *Success: added new records in list process must be killed* * *Fail: Some missing process information*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list process real time which show student computer process real time.* | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* | | *2* | *Select the process row which IT want to Add* | *Parse process data to process information format* | | *3* | *IT send add to process must kill command* |  | | *4* |  | *Add new Process to Process must killed. System check if process information is missing*  *[Exception]* |   **Exceptions:**   |  |  |  | | --- | --- | --- | | **Step** | **Cause** | **System Response** | | *1* | *Process information is missing* | *Show message to IT client* |   **Relationships:**   * *Get Realtime Processes*   **Business Rules:**   * *When student start the Exam Client up, IT Managers can get the real time processes of that student, then he can add any of those processes to the must kill list.* * *Processes added into the process must kill list can be killed instantly when the Exam Client start up.* * *Process must be inside the student’s real time process list.* | | | |

Table 15 Add Must Be Killed Process to List

#### <IT> Add to List Exception Process

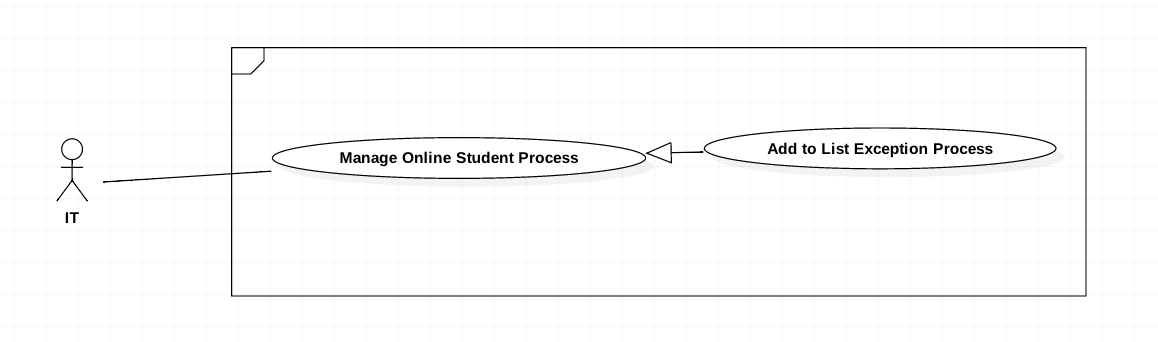


Figure 11 Add to List Exception Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.04** | | | |
| **Use Case No.** | UC\_IT04 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add to List Exception Process | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   ***Summary:***   * *Allows IT add process from real time process to List Exception Process*   ***Goal:***   * *Add a new exception process.*   **Trigger:**   * *When IT request add new process exception command.*   **Preconditions:**   * *IT scan student computers ‘s running processes.*   **Post Conditions:**   * *Success: added new records in list exception processes* * *Fail: Process information is missing*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list process real time which show student computer process real time.* | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* | | *2* | *Select the process row which IT want to Add* | *Parse process data to process information format* | | *3* | *IT send add to process exception command* |  | | *4* |  | *Add new Process to Process must killed. System check if process information is missing*  *[Exception]* |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | *1* | *Process information is existed* | *Notify to IT Client* |   **Relationships:**   * *Get Realtime Processes*   **Business Rules:**   * *When student start the Exam Client up, IT Managers can get the real time processes of that student, then he can add any of those processes to the exception list.* * *Processes in the exception list doesn’t killed when student start the Exam Client.* * *Process must be inside the student’s real time process list.* | | | |

Table 16 ADD TO LIST EXCEPTION PROCESS

#### <IT> Kill Process

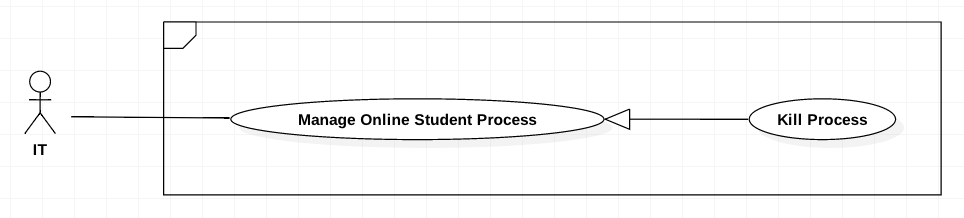


Figure 12 Kill Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.05** | | | |
| **Use Case No.** | UC\_IT05 | **Use Case Version** | 2.0 |
| **Use Case Name** | Kill Process | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *When the IT want to remove process on student computer, IT can click X button to remove processes on student computer.*   **Goal:**   * *IT can remove process which must be killed they scanned for preventing cheating.*   **Triggers:**   * *IT call command to server.*   **Preconditions:**   * *IT must scan student computer ‘s processes.* * *IT must record processes which they defined cheating processes.*   **Post Conditions:**   * *Success: Process is killed on student computer.* * *Fail: Notify an error to IT*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list process real time which show student computer process real time.* | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* | | *2* | *Select the process row which IT want to Kill* | *Parse process data to process information format* | | *3* | *IT send Kill Process command* |  | | *4* |  | *Process selected will kill on all student computer if it existed. System check if process information is missing*  *[Exception]* |   **Relationships:**   * *Get Realtime Processes*   **Business Rules:**   * *When student start the Exam Client up, IT Managers get the real time processes of that student.* * *Processes of a student in can be selected and can be killed manually.* * *Process must be inside of the student’s real time process list.* | | | |

Table 17 Kill Process

#### <IT> Get Realtime Process

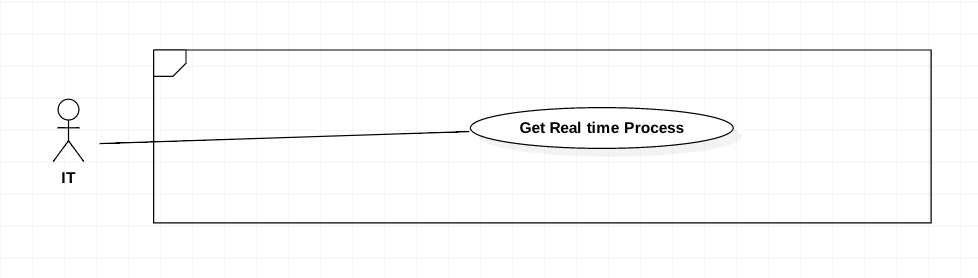


Figure 13 <IT> Get Realtime Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.06** | | | |
| **Use Case No.** | UC\_IT06 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get Realtime Processes | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT can scan all process running on Student computer.*   **Goal:**   * *IT can scan all of processes which are running in student computers for preventing cheating.*   **Triggers:**   * *IT send get real time process command.*   **Preconditions:**   * *Examtool must be running in student computer.*   **Post Conditions:**   * *Success: Get list of running processes.* * *Fail: Show error message*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list process real time which show student computer process real time.* | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* |   **Business Rules:**   * *When student running the ExamTool, IT Managers can get all processes running on that computer.* | | | |

Table 18 <IT> GET REALTIME PROCESS

#### <IT> Search Realtime Process

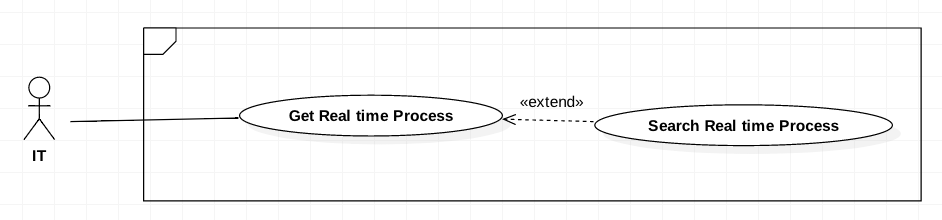


Figure 14 <IT> Search Realtime Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.07** | | | |
| **Use Case No.** | UC\_IT07 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search Realtime Process | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT search processes in list of scanned running processes.*   **Goal:**   * *IT can find processes which are running in student computers for preventing cheating.*   **Preconditions:**   * *Student Client must be running in student computer.* * *IT got a list of scanned running processes.*   **Post Conditions:**   * *Success: receive process information if process exist.* * *Fail: Show error message*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list process real time which show student computer process real time.* | *The system will be displayed list process must kill with information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* | | *2* | *IT input search value* | *The system matching searching value with process information:*   * *ProcessName* * *ApplicationName* * *ProcessPath* |   **Relationships:**   * *Get Realtime Processes*   **Business Rules:**   * *When student running the ExamTool, IT Managers can search any Process with 3 information running on Student computer.* * *Processes must be inside the real time process list.* | | | |

Table 19 <IT> SEARCH REALTIME PROCESS

#### <IT> Get Student Screen

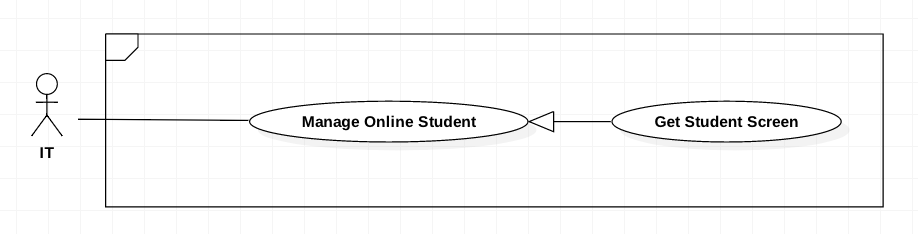


Figure 15 <IT> Get Student Screen

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.08** | | | |
| **Use Case No.** | UC\_IT08 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get student screen | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * IT   **Summary:**   * *This use case allows IT can view student screen.*   **Goal:**   * *IT can prevent cheating when trace back images recorded student screens.*   **Preconditions:**   * *ExamTool must be running in student computer.*   **Post Conditions:**   * *Success: Show student screen on view* * *Fail: show error message*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list student online and login to ExamTool* | *The system run get all student online with ExamTool command.* | | *2* |  | *Show list student online with ExamTool with information:*   * *Student User* * *Student Name* * *ConnectionId* | | *3* | *IT select student want to get screen.* | *The system run get Student screen command with connection Id parameter.* | | *4* |  | *The system request to ExamTool have this connection Id and response continuous capture image.* | | *5* |  | *The system show continuous capture image to screen.* |   **Business Rules:**   * *After students logged into the ExamTool, IT Manager can get the list of those student who is online.* * *IT Managers choose a student to watch their current screen in real time.* * *IT Managers can only choose students in the online list.* | | | |

Table 20 <IT> GET STUDENT SCREEN

#### <IT> Submit Student Test Exam

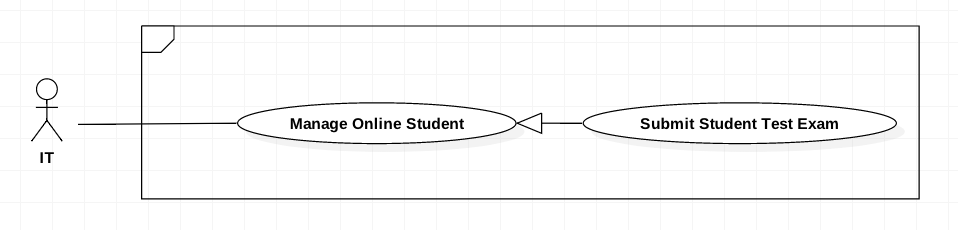


Figure 16 <IT> Submit Student Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.09** | | | |
| **Use Case No.** | UC\_IT09 | **Use Case Version** | 2.0 |
| **Use Case Name** | Submit Student Test Exam | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT can submit last student exam saved on local any time.*   **Goal:**   * *IT can prevent losting exam result when student client is crashed or get error in submiting process.*   **Preconditions:**   * *ExamTool must be running in student computer.* * *Student must login in ExamTool*   **Post Conditions:**   * *Success: Submit last test exam to Server* * *Fail: Show error message.*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list student online and login to ExamTool* | *The system run get all student online with ExamTool command.* | | *2* |  | *Show list student online with ExamTool with information:*   * *Student User* * *Student Name* * *ConnectionId* | | *3* | *IT select student want to submit test exam* |  | | *4* | *IT goes to submit student test exam screen and run submit student exam command* | *The system request to ExamTool with this connectionId to submit last saved test exam.* | | *5* |  | *ExamTool Submit last saved test exam to server.*  *[Exception]* |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | *1* | *Last Test Exam already submit successed.* | *Show error message* |   **Business Rules:**   * *IT can submit the student test exam at any time during the test.* * *Student must logged and running ExamTool.* * *The last test exam never ever submitting successed.* * *The last test exam must saved on student computer.* * *IT can submit the student test exam saved locally in their computer if they have problem with internet connection.* | | | |

Table 21 <IT> Submit Student Test Exam

#### <IT> Search Online Student

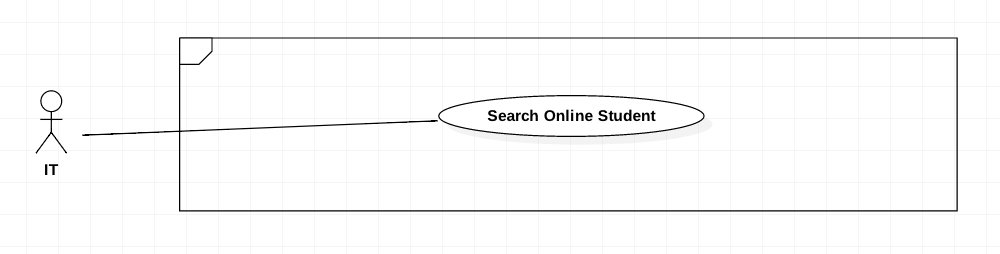


Figure 17 <IT> Search Online Student

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.10** | | | |
| **Use Case No.** | UC\_IT10 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search online student | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows IT to find the student who using with ExamTool.*   **Goal:**   * *IT can check student who doing test.* * *Can find specify student to get screen, kill process or submit test exam.*   **Preconditions:**   * *Student Client must be running in student computer.*   **Post Conditions:**   * *Success: Get student status.* * *Fail: Notify an error to IT*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *IT goes to list student online and login to ExamTool* | *The system run get all student online with ExamTool command.* | | *2* |  | *Show list student online with ExamTool with information:*   * *Student User* * *Student Name* * *ConnectionId* | | *3* | *IT input search value* | *The system matching text between search value and online student information:*   * *Student User* * *Student Name* * *ConnectionId* |   **Business Rules:**   * *After students logged into the ExamTool, IT Manager can get the list of those student who is online.* * *Student must logged by ExamTool,* * *IT Managers can search students on that list by Student Name or Student Code.* * *Can only search students who currently logged into the Exam Client*. | | | |

Table 23 <IT> SEARCH ONLINE STUDENT

#### <Training Department Staff> Add new Semester

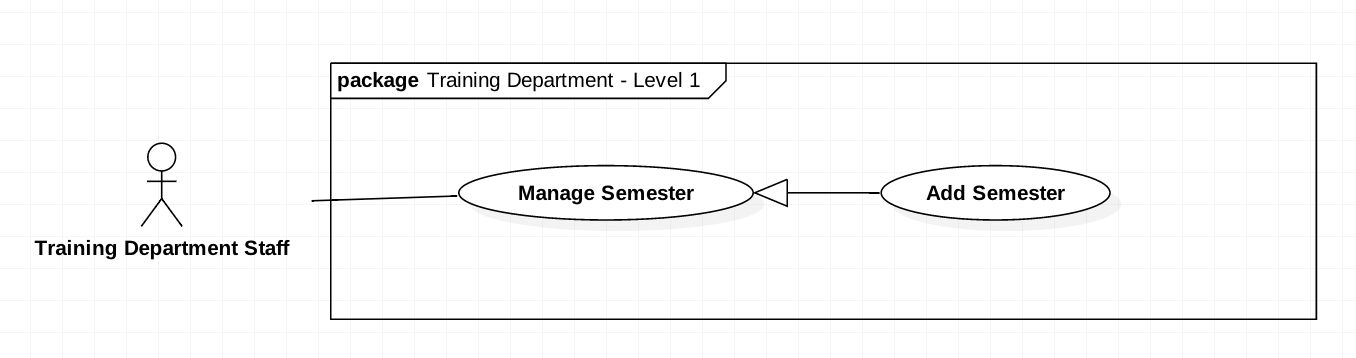


Figure 18 <Training Department Staff> Add new Semester

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ TrainingDepartmentStaff.11** | | | |
| **Use Case No.** | UC\_ Training Department Staff 11 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add new Semester | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| 1. **Actor:**    * + *Training Department Staff.* 2. **Summary:**    * + *This use case allows Training Department Staff to add a new semester in order to create exams.* 3. **Goal:**    * + *Admin* *can create a new semester with start date and end date so that exams can be created in that semester.* 4. **Triggers:**    * + *Training Department Staff sends a create command to create a new semester.* 5. **Preconditions:** 6. **Post Conditions:**    * + *Success: The new semester will be saved.*      + *Fail: Show error message* 7. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Actor sends a create command to create a new semester.* | *The system requires information of the semester:*   * + *Semester code* | | *2* | *Actor input information* |  | | *3* | *Actor sends a command to send* |  |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *The actor does not send a command to add medicine.* | *The system will not display semester information.* | | *2* | *The actor sends a command to cancel.* | *The system will close the create semester view.* |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *Semester record has existed* | *Show message to notify the actor that the semester has existed in the system.* | | *2* | *The actor doesn’t fill the semester code* | *Show message to notify the actor that the semester code is required.* |   **Relationships: Manage Semester**  **Business Rules:**   * + *Semester name cannot be duplicated*   + *End Date cannot before Start Date*   + *Training Department Staff log into the system, then Staff can create a new Semester with Code, Start Date and End Date.*   + *Training Department Staff need to create a new Semester to create Exam and test exam in this Semester.* | | | |

Table 24 <TRAINING DEPARTMENT STAFF> ADD NEW SEMESTER

#### <Training Department Staff> Get Semester



Figure 19 <Training Department Staff> Get Semester Detail

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ TrainingDepartmentStaff.12** | | | |
| **Use Case No.** | UC\_ TrainingDepartmentStaff12 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get semester Detail | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | **Normal** |
| 1. **Actor:**    * *Training Department Staff* 2. **Summary:**    * *This use case allow Tranning Department Staff can view list all semester.* 3. **Goal:**    * *Training Department Staff can see the details of the semester’s information* 4. **Triggers:**    * *The Training Department Staff sends a command to view the patient’s* information. 5. **Preconditions: N/A** 6. **Post conditions:**    * *Success: All semesters show up.*    * *Fail: Show error message.* 7. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Actor sends a command to view semesters.* | *The system requires information of the semester:*   * + *Semester code* | | *2* | *Actor input information* |  | | *3* | *Actor sends a command to send* |  |  1. **Alternative Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *The actor does not send a command to add medicine.* | *The system will not display semester information.* | | *2* | *The actor sends a command to cancel.* | *The system will close the create semester view.* |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *Semester record has existed* | *Show message to notify the actor that the semester has existed in the system.* | | *2* | *The actor doesn’t fill the semester code* | *Show message to notify the actor that the semester code is required.* |   **Relationships: N/A**  **Business Rules:**  - *Training Department Staff log into the system, then go to the semester page to get the semester from the database.*  *- Semester must added before.* | | | |

Table 25 <Training Department Staff> Get Semester Detail

#### <Manager> Public Test Exam

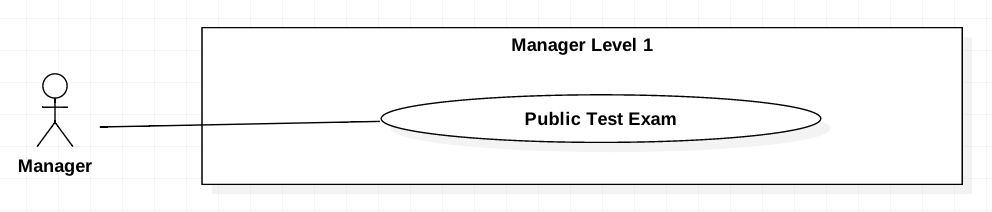


Figure 20 <Manager> Public Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Manager.13** | | | |
| **Use Case No.** | UC\_ Manager 13 | **Use Case Version** | 2.0 |
| **Use Case Name** | Public Test Exam | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + *Manager*   **Summary:**   * + *This use case allows managers approve test exam to available.*   **Goal:**   * + *Manager can approve exam to available by click the publish button*   **Triggers:**   * + *Leader send publish command*   **Preconditions:**   * + *Login the system with “Manager” role.*   **Post Conditions:**   * + *Success: Test exam will be changed to “Available”*   + *Fail: Test exam still be “Approve” and show message error*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | *1* | *Manager goes to List Exam Approved Test Exam* | *The system will display list test exam with information:*   * + *Approver*   + *Teacher mark*   + *Course* | | *2* | *Manager send public test exam command* | *The system change test exam status to public.*   * 1. *Update status test exam on interface* |  1. Alternative Scenario:  |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | *1* | *Leader sends a command to get courses to follow name or code in the search field* | *Show result not found on the screens* |   **Exceptions: N/A.**  **Relationships: N/A.**  **Business Rules:**   * *Manager can public or off the test exam* * *When public test exam the Training Department Staff can create exam by this test exam.* * *If Manager turn off Test exam, the exam can’t create with this test exam.* | | | |

Table 26 <MANAGER> APPROVE AVAILABLE TEST EXAM

#### <Leader> Search Chapter

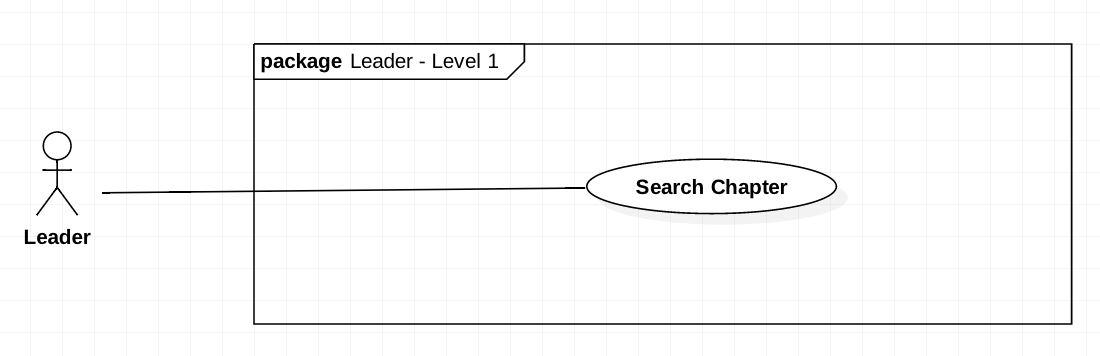


Figure 21 <Leader> Search Chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Leader.14** | | | |
| **Use Case No.** | UC\_Leader 14 | **Use Case Version** | 2.0 |
| **Use Case Name** | SearchChapter | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Leader*   **Summary:**   * *This use case allows search for chapters.*   **Goal:**   * *Leader can search for chapters by name or by code.*   **Triggers:**   * *Leader send search command*   **Preconditions:**   * *Login the system with Leader role.*   **Post Conditions:**   * *Success: Chapters will be found or not be found* * *Fail: N/A*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Leader input name or code in the search field* |  | | *2* | *Leader sends a command to get chapters to follow name or code in the search field* | *Show chapters have been searching on the chapter list:*   * + *Code*   + *Course*   + *Level*   + *Mark* |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Leader sends a command to get chapters to follow name or code in the search field* | *Show result not found on the screens* |   **Exceptions: N/A.**  **Relationships: N/A.**  **Business Rules:**   * 1. *Leader will get the chapter list.*   2. *Leader will input chapter information. Just input name or code, cannot input both of them.*   3. *The result after searching will be shown with an approximate code or name.* | | | |

Table 27 <Leader> Search Chapter

#### <Leader> Add chapter

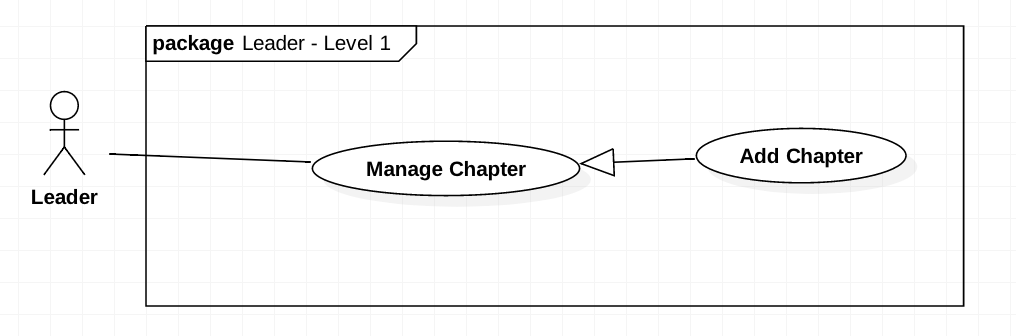


Figure 22 Add chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Leader.15** | | | |
| **Use Case No.** | UC\_Leader 15 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add Chapter | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * 1. *Leader*   **Summary:**   * 1. *This use case allows Leader to add chapters.*   **Goal:**   * 1. *Add new chapter for specify course.*   **Triggers:**   * 1. *Leader send an add chapter by form command.*   **Preconditions:**   * 1. *Login the system with Leader role.*   **Post Conditions:**   * + - *Success: Chapters will be added.*     - *Fail: Chapters will not be added.*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | 1 | Leader goes to add chapter view | The system require to fill information:   * + *Chapter*   + *Level*   + *Mark*   + *Chapter Content*   + *Options* | | 2 | Leader send a save command to save the chapters locally |  | | 3 |  | The system show response message and display new list chapter.  [Exception] |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *The chapter info is missing* | *Show message to refill form to add new chapter.* |   **Alternative Scenario: N/A**  **Relationships: Manage Chapter**  **Business Rules:**   * + - *Leader log into the system, go to the Course page and add new chapter.*     - *Leader must fill full chapter content.*     - *Chapter Content cannot duplicate to others.* | | | |

Table 28 Add chapter

#### <Leader> Delete chapter

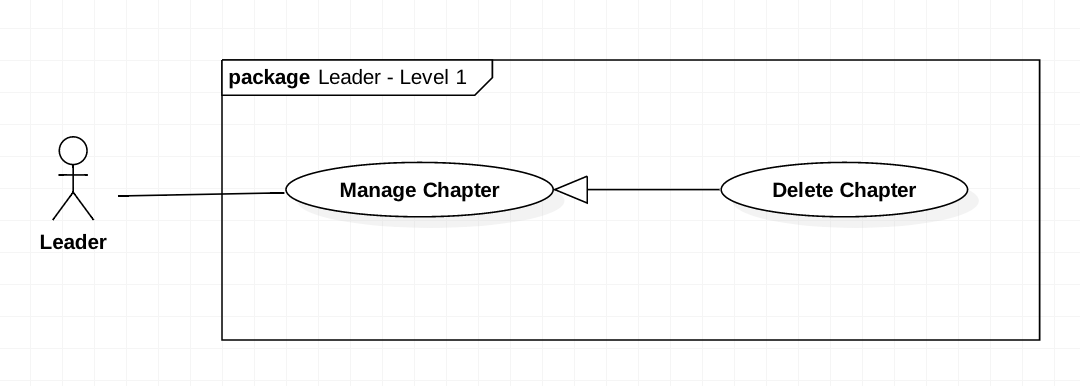


Figure 23 Delete chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Leader.16** | | | |
| **Use Case No.** | UC\_Leader16 | **Use Case Version** | 2.0 |
| **Use Case Name** | Delete Chapter | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + *Leader*   **Summary:**   * + *This use case allows Leader to remove chapters.*   **Goal:**   * + *Remove the chapter from list chapter of this course*   **Triggers:**   * + *Leader send a delete chapter command.*   **Preconditions:**   * + *Login the system with Leader role.*   **Post Conditions:**   * + *Success: Chapters will be added.*   + *Fail: Chapters will not be added and show error message*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | |  | *Leader goes to list chapter* | *The system will display list chapter with information:*   * + *Chapter*   + *Level*   + *Mark*   + *Chapter Content*   + *Options* | |  | *Leader select chapter want to remove* |  | | *3* | *Leader send a delete chapter command.* | *The system will show a popup notify if that chapter has been deleted successfully* |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: Manage Chapter**  **Business Rules:**   * *Leader can remove chapter no longer available by log into the system and choose course contain the chapter.* * *That chapter must be available in the content.* | | | |

Table 29 Delete chapter

#### <Leader> Edit chapter

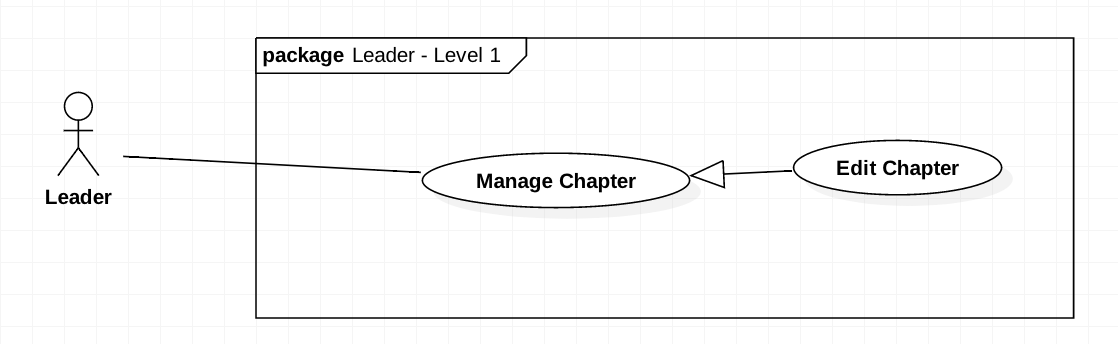


Figure 24 <Leader> Edit chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Leader.18** | | | |
| **Use Case No.** | UC\_Leader18 | **Use Case Version** | 2.0 |
| **Use Case Name** | Edit Chapter | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + - *Leader*   **Summary:**   * + - *This use case allows Leader to edit chapters*.   **Goal:**   * + - *Edit chapter with new content.*   **Triggers:**   * + - *Leader send an edit chapter command.*   **Preconditions:**   * + - *Login the system with Leader role.*   **Post Conditions:**   * + - *Success: Chapters will be edited.*     - *Fail: Chapters will not be edit and show message error*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Leader goes to List chapter view* | *The system will display list chapter with information:*   * + *Chapter*   + *Level*   + *Mark*   + *Chapter Content*   + *Options* | | *2* | *Leader fill the form* | *The system requires Leader to input:*   * + *Chapter*   + *Level*   + *Mark*   + *Chapter Content*   + *Options* | | *3* | *Leader send a save command to save the chapters locally* | *[Exception]* | | *4* | *Leader send save command* | *The system will run save change command to save all changed* |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *The chapter info is missing* | *Show message to fill full form,* |   **Alternative Scenario: N/A**  **Relationships: Manage Chapter**  **Business Rules:**   * + - *Leader will get list chapter*     - *Leader can change multi chapter name per time by changing some chapter and submit last time.*     - *Chapter Content cannot duplicate to others.*     - *Leader log into the system, go to the Course page and choose a chapter available on the screen to edit.* | | | |

Table 30 <Leader> Edit chapter

#### <Teacher> Search Question



Figure 25 <Teacher> Search Question

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.19** | | | |
| **Use Case No.** | UC\_Teacher 19 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search Question | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Teacher*   **Summary:**   * *This use case allows search for questions.*   **Goal:**   * *Finding the question with matching code and question content.*   **Triggers:**   * *Teacher send search command*   **Preconditions:**   * *Login the system with teacher role.*   **Post Conditions:**   * *Success: Questions will be found or not be found* * *Fail: N/A*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher will go to list question page* | *The system will display list question information at specify page:*   * + *Code*   + *Course*   + *Mark*   + *Level* | | *1* | *Teacher input name or code in the search field* |  | | *2* | *Teacher sends a command to get questions to follow name or code in the search field* | *The system run command and show questions have been searching on the question list:*   * + *Code*   + *Course*   + *Level*   + *Mark* |  1. **Alternative Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher sends a command to get questions to follow name or code in the search field* | *Show result not found on the screens* |   **Exceptions: N/A.**  **Relationships: N/A.**  **Business Rules:**   * *The teacher will get list question of the course department* * *The teacher can edit or remove question by select question and send edit or remove command.* * *The teacher will input question information. Just input name or code, cannot input both of them.* * *The result after searching will be shown with an approximate code or name.* * *Teacher log into the system, go to the Course page, Manage Question tab and can search question by content or by code.* | | | |

Table 31 <Teacher> Search Question

#### <Teacher> Add Question by Form



Figure 26 <Teacher> Add Question by Form

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.20** | | | |
| **Use Case No.** | UC\_Teacher 20 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add Question by Form | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * 1. Teacher   **Summary:**   * 1. This use case allows teacher to add questions by form.   **Goal:**   * 1. *Teacher can add question by form.*   **Triggers:**   * 1. *Teacher send an add question by form command.*   **Preconditions:**   * 1. *Login the system with teacher role.*   **Post Conditions:**   * 1. *Success: Questions will be added.*   2. *Fail: Questions will not be added.*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher fill the form* | *The system requires teacher to input:*   * + *Chapter*   + *Level*   + *Mark*   + *Question Content*   + *Options* | | *2* | *Teacher send a save command to save the questions locally* |  | | *3* | *Teacher send a send command add question.* | *The system will run command and question and comeback to question list.*  *[Exception]* |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *Teacher not fill full question info* | *Show message to fill full form.* | | *2* | *Teacher not choose write answer for question.* | *Show message require answer* |  1. **Alternative Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher add question with content matching >80% question already in database* | *Show warning may question already in database and confirm continuous to add or cancel action.* |   **Relationships: Add Question**  **Business Rules:**   * + *Teacher must fill in all inputs of the form except the Learning Outcome.*   + *The form first must be saved locally, after that it can be submitted and saved to the database.*   1. *If Question content duplicate other. The system will warning and teacher can remove or continuous add, edit question.*   2. *Total Mark must be 100%.*   3. *Each question must have at least 2 options and at least 1 answer.*   4. *Teacher log into the system, go to Course page, Manage Question tab, Click the Import Question button and Add Question button to add new question by form.* | | | |

Table 32 <TEACHER> ADD QUESTION BY FORM

#### <Teacher> Add Question by Import File

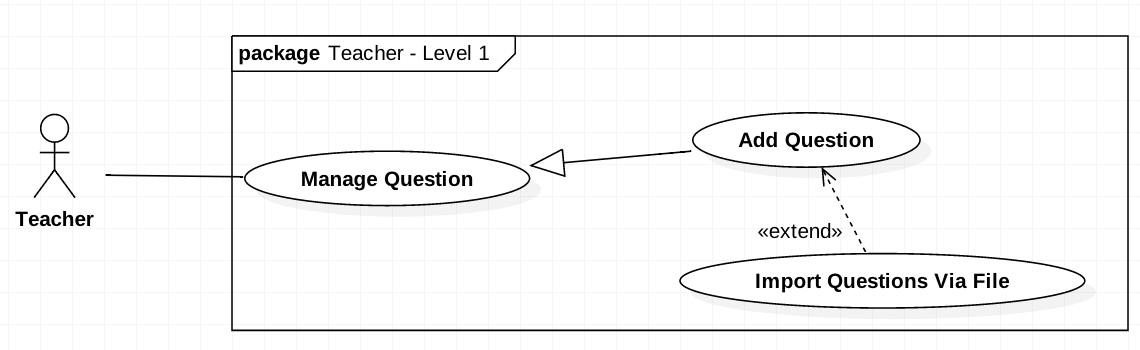


Figure 27 <Teacher> Add Question by Import File

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.21** | | | |
| **Use Case No.** | UC\_Teacher 21 | **Use Case Version** | 2.0 |
| **Use Case Name** | Add Question by Import File | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Teacher*   **Summary:**   * *This use case allows teacher to add questions by uploading file.*   **Goal:**   * *Upload multiple question list exported by moodle (old system)*   **Triggers:**   * *Teacher send an add questions command.*   **Preconditions:**   * *Login the system with teacher role.*   **Post Conditions:**   * *Success: Questions will be added.* * *Fail: Questions will not be added.*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | |  |  |  | | *1* | *Teacher upload the file* | *The system requires teacher to upload a file with GIFT format or file extensions: .doc,.xml* | |  |  | *System run command to validate and check duplicate question.* | | *2* | *Teacher review the uploaded questions* | *Show question format status:*   * + *Success*   + *Warning*   + *Error* | | *3* | *Teacher send a submit command to save the questions to the database* | *System run add questions to add question list* |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: Add Question**  **Business Rules:**   * + - *Teacher can add multiple question by import file.*     - *From the result warning teacher can edit question of cancel upload duplicate question.*     - *Teacher must upload the required file extension and format exported from Moodle.*     - *Teacher log into the system, go to Course page, Manage Question tab, Click the Import Question button and Upload File button to add new question by file.* | | | |

Table 33 <Teacher> Add Question by Import File

#### <Teacher> Remove Questions

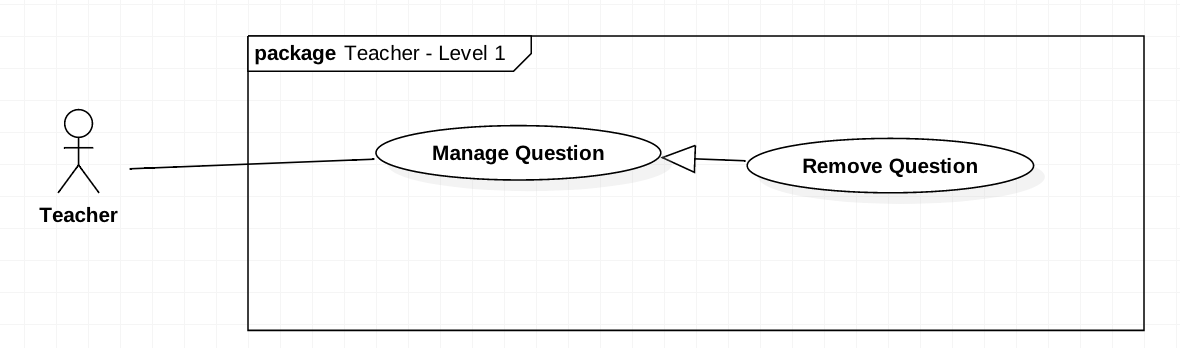


Figure 28 <Teacher> Remove Questions

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.22** | | | |
| **Use Case No.** | UC\_Teacher 22 | **Use Case Version** | 2.0 |
| **Use Case Name** | Remove Questions | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + *Teacher*   **Summary:**   * + *This use case allows teacher to remove questions.*   **Goal:**   * + *Remove questions from question list*   **Triggers:**   * + *Teacher send a delete question command.*   **Preconditions:**   * + *Login the system with teacher role.*   **Post Conditions:**   * + *Success: Questions will be removed.*   + *Fail: Questions will not be removed and show error message*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher goes to list question list* | *The system will display list question information:*   * + *Course*   + *Mark*   + *Level* | | *2* | *Teacher choose delete question* | *The system will show a warning popup to ensure delete action again.* | | *3* | *Teacher confirm remove question in popup.* |  | | *4* | *Teacher send a delete question command.* | *The system will show a popup notify that question has been deleted successfully* |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: Manage Question**  **Business Rules:**  - *Teacher log into the system, go to Course page, Manage Question tab, choose a question to remove.*  *- Question must be available in the Manage* *Question tab* | | | |

Table 34 <Teacher> Remove Questions

#### <Teacher> Edit Questions

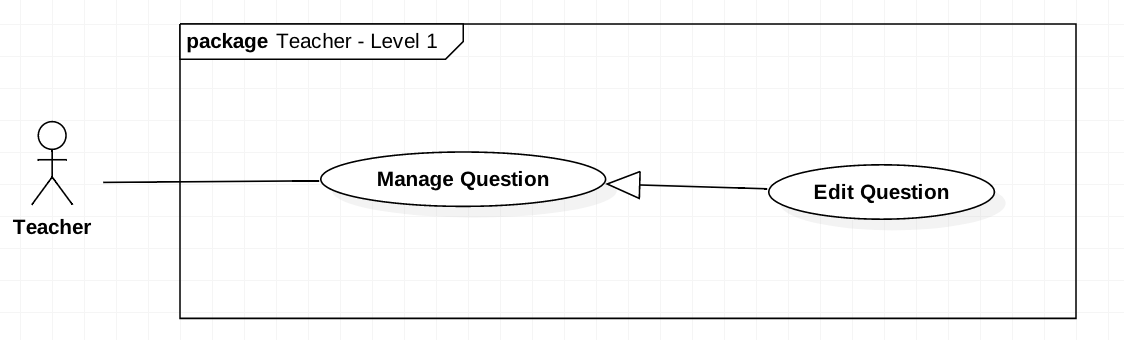


Figure 29 <Teacher> Edit Questions

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.23** | | | |
| **Use Case No.** | UC\_Teacher 23 | **Use Case Version** | 2.0 |
| **Use Case Name** | Edit Questions | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Teacher*   **Summary:**   * *This use case allows teacher to edit questions.*   **Goal:**  *Edit question content, answer, percent when student check wrong option, level... of question*  **Triggers:**   * *Teacher send an edit question command.*   **Preconditions:**   * *Login the system with teacher role.*   **Post Conditions:**   * *Success: Questions will be edited.* * *Fail: Questions will not be edit and show message error*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher goes to list question view* | *The system will display list question information:*   * + *Course*   + *Mark*   *Level* | | *2* | *Teacher choose to view detail question* | *The system will display detail information of this question:*   * + *Question code*   + *Question content*   + *Option*   + *Learning Outcome*   + *Chapter*   + *Mark*   + *Percent*   + *Level*   + *Right answer* | | *3* | *Teacher edit question information* | *The system require fill full info except learning outcome.* | | *4* | *Teacher send a submit command to save the questions.* | *The system run save question command to save and comeback to list question.* |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: Manage Question**  **Business Rules:**   * *Total Mark must be 100%.* * *Each question must have at least 2 options and at least 1 answer.* * *Teacher log into the system, go to Course page, Manage Question tab, choose a question to edit content and answers.* * *Question must be available in the Manage Question tab* | | | |

Table 35 <Teacher> Edit Questions

#### <Teacher> Get List Test Exam

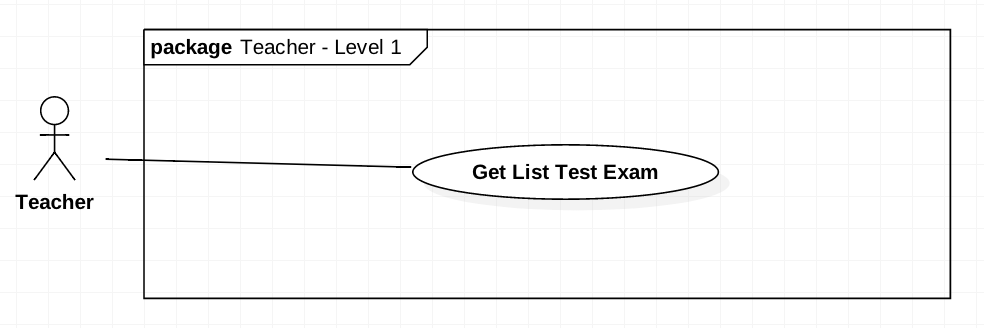


Figure 30 <Teacher> Get List Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.24** | | | |
| **Use Case No.** | UC\_Teacher 24 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get List Test Exam | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Teacher*   **Summary:**   * *This use case allows teacher to get list test exam assigned by training department staff to that teacher.*   **Goal:**   * *Get list test exam assigned to approve.* * *Teacher can approve or reject in that list test exam.*   **Triggers:**   * *Teacher send a get list exam command.*   **Preconditions:**   * *Login the system with teacher role.*   **Post Conditions:**   * *Success: List of test exam will be shown.* * *Fail: List of test exam will not be shown.*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher goes to Approve Test Exam view of Semester which teacher want to get.* | *The system will display list test exam assigned to that teacher with information:*   * + *Course*   + *Test Exam status (Not available, Edited, Taken)*   + *Time and duration to take.* | | *2* | *Teacher send a command to get list of exam.* | *Show list test exam assigned with information:*   * + *Course*   + *Test Exam status (Not available, Edited, Taken)*   + *Time and duration to take.* |   **Alternative Scenario: N/A**  **Exceptions: N/A**  **Relationships: N/A**  **Business Rules:**   * *Teacher can get list test exam assigned.* * *Teacher can’t edit get test exam already approved.* * *Teacher can take an exam to approve, reject or edit that test exam.* | | | |

Table 36 <Teacher> Get List Test Exam

#### <Teacher> Approve Test Exam

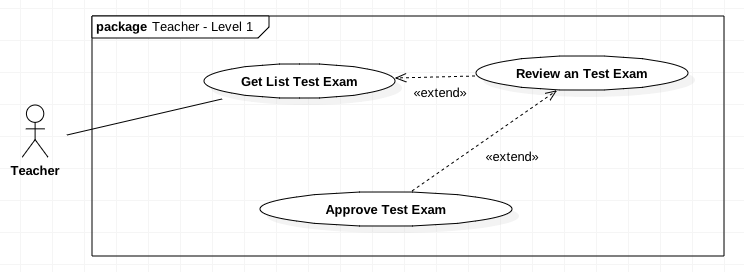


Figure 31 <Teacher> Approve Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Teacher.25** | | | |
| **Use Case No.** | UC\_Teacher 25 | **Use Case Version** | 2.0 |
| **Use Case Name** | Approve Test Exam | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *Teacher*   **Summary:**   * *This use case allows teacher to approve an exam assigned to that teacher.*   **Goal:**   * *Approve test exam quality to storage in test exam bank*   **Triggers:**   * *Teacher send an approve exam command.*   **Preconditions:**   * *Login the system with teacher role.*   **Post Conditions:**   * *Success: Exam status will be changed to “Approved”.* * *Fail: Exam will still be “Waiting”.*  1. **Main Success Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher goes to Approve Test Exam View* | *The system will display list test exam information assigned to that teacher:*   * + *Course*   + *Number of questions*   + *Duration* | |  | *Teacher send command to view detail Test Exam* | *The system will display list question and require teacher take that exam.* | | *3* | *Teacher take an exam* |  | | *4* | *Teacher submit test exam* | *The system will calculate mark and change Test Exam status to taken and display Test Exam result to interface with information:*   * + *Mark*   + *Test Exam Result* | | *5* | *Teacher send command approve/ reject test exam.*  *[Alternative Scenario 1]* | *The system will change Test Exam to approved or rejected and come back display list exam assigned to approve.* |  1. **Alternative Scenario:**  |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Teacher send command edit questions* | *The system will require teacher to retake exam* |   **Exceptions: N/A**  **Relationships: Extend “Review an test exam”**  **Business Rules:**   * *Teacher must approve test exam in range time setup by training department staff* * *Teacher can only approve exams which are assigned to his/her.* * *Teacher must take a test exam to approve the test exam.* * *With any changed (question, option, …) Teacher must come back to take exam process.* * *Teacher cannot approve exams which are already approved.* * *The result of test exam will be display on Public test exam view of Manager view.* * *Teacher log into the system, go to the Review/Approve Exam page and choose a test exam to submit for approving.* | | | |

Table 37 <Teacher> Approve Test Exam

#### <Training Department Staff> Generate Test Exam

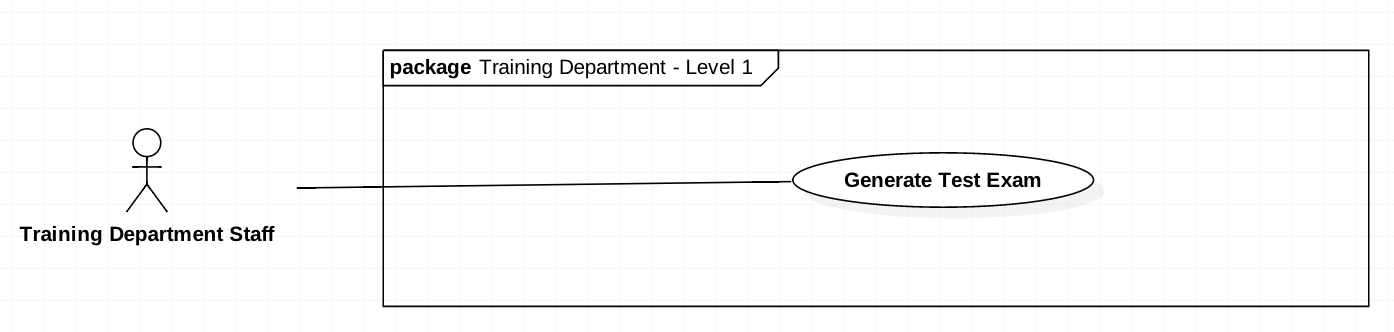


Figure 31 <Teacher> generate Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Training Department Staff.25** | | | |
| **Use Case No.** | UC\_ Training Department Staff 25 | **Use Case Version** | 2.0 |
| **Use Case Name** | Generate Test Exam | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + *Training department staff*   **Summary:**   * + *This use case allows training department staff to generate test exam*   **Goal:**   * + *Generate new Test Exam to Test Exam bank.*   **Triggers:**   * + *Training Department Staff call Generate Test Exam command*   **Preconditions:**   * + *Login the system with “Training department staff” role.*   **Post Conditions:**   * + *Success: Generate new Test Exam*   + *Fail: Test exam not be generated and show message error*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Manager click the Approve/Reject Exam button in the menu* |  | | *2* | *Manager click the Approve/Reject Exam tab* |  | | *3* | *Teacher select the semester* |  | | *4* | *Teacher send a command to get list of exam* | *Show list of exam with:*  *- Course*  *- Start Date*  *- End Date*  *- Duration*  *- Status*  *- Actions* | | *5* | *Teacher click “More Detail” button of an exam in the Actions column* |  | | *6* | *Teacher click the “Submit” button* |  | | *7* | *Teacher click the “Approve” button* |  | | *8* | *Teacher send an approve exam command* |  |   **Alternative Scenario: N/A**  **Exceptions: N/A.**  **Relationships: N/A.**  **Business Rules:**   * 1. *The Leader will input course information. Just input name or code, cannot input both of them.*   2. *The result after searching will be shown with an approximate code or name.*   3. *The result of searching will display on result list.* | | | |

Table 38 <Training department staff> Generate Test Exam

#### <Training Department Staff> Assign Approver

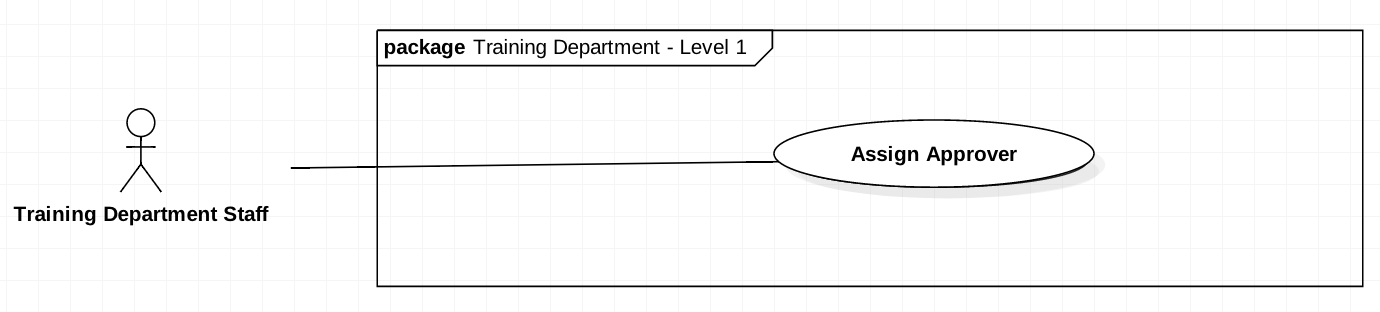


Figure 31 < Training Department Staff > Approve Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_Training Department Staff.25** | | | |
| **Use Case No.** | UC\_ Training Department Staff 25 | **Use Case Version** | 2.0 |
| **Use Case Name** | Assign Approver | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * + *Training department staff*   **Summary:**   * + *This use case allows training department staff to assign the teacher to approve test exam.*   **Goal:**   * + *Assign Test Exam to teacher who approve that test exam*   **Triggers:**   * + *Training Department Staff call Set approver command*   **Preconditions:**   * + *Login the system with “Training department staff” role.*   **Post Conditions:**   * + *Success: Assign test exam to approver*   + *Fail: Can’t assign test exam to approve*   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | **Step** | **Actor Action** | **System Response** | | *1* | *Training Department Staff goes to Assign Approver view* | *System require complete information:*   * + *Course*   + *Test Exam*   + *Start Date to approve*   + *End Date to take Exam*   + *Approver* | | *2* | *Training Department Staff full fill information* |  | | *3* | *Teacher run Assign Approver command* | *The system run Assign Approver command*  *[Exception]* |   **Exception:**   |  |  |  | | --- | --- | --- | | **No** | **Cause** | **System Response** | | *1* | *Staff not full fill form* | *Show error message* |   **Alternative Scenario: N/A**  **Relationships: N/A.**  **Business Rules:**   * 1. *Training Department Staff can assign one Teacher to approve the test exam*   2. List Test Exam can assign must already public by manager   3. Start date must be less than end date | | | |

Table 38 <Training department staff> Assign Approver

## Software System Attribute

### Usability

* + UI: Font size: 8px – 30px.

Color: Light Red, Light Green, Blue, White, Light Gray

* 1. Background: White
  2. Font size: 12dp – 18dp  
     Color: While, Blue, Light Grey, Light Red Background: White

### Reliability

* 1. All appointment is recorded, not missing a single one. Web application never crash in any circumstance.

### Availability

* 1. System replies in maximum 5 seconds.

### Security

* 1. Web application have multiple role and each role has a specific permission to interact with.

### Maintainability

* 1. All server software, web application is divided into separated modules for easy maintain.

### Portability

* 1. Web application can be run on Chrome browser version 42 or later.

## Conceptual Diagram

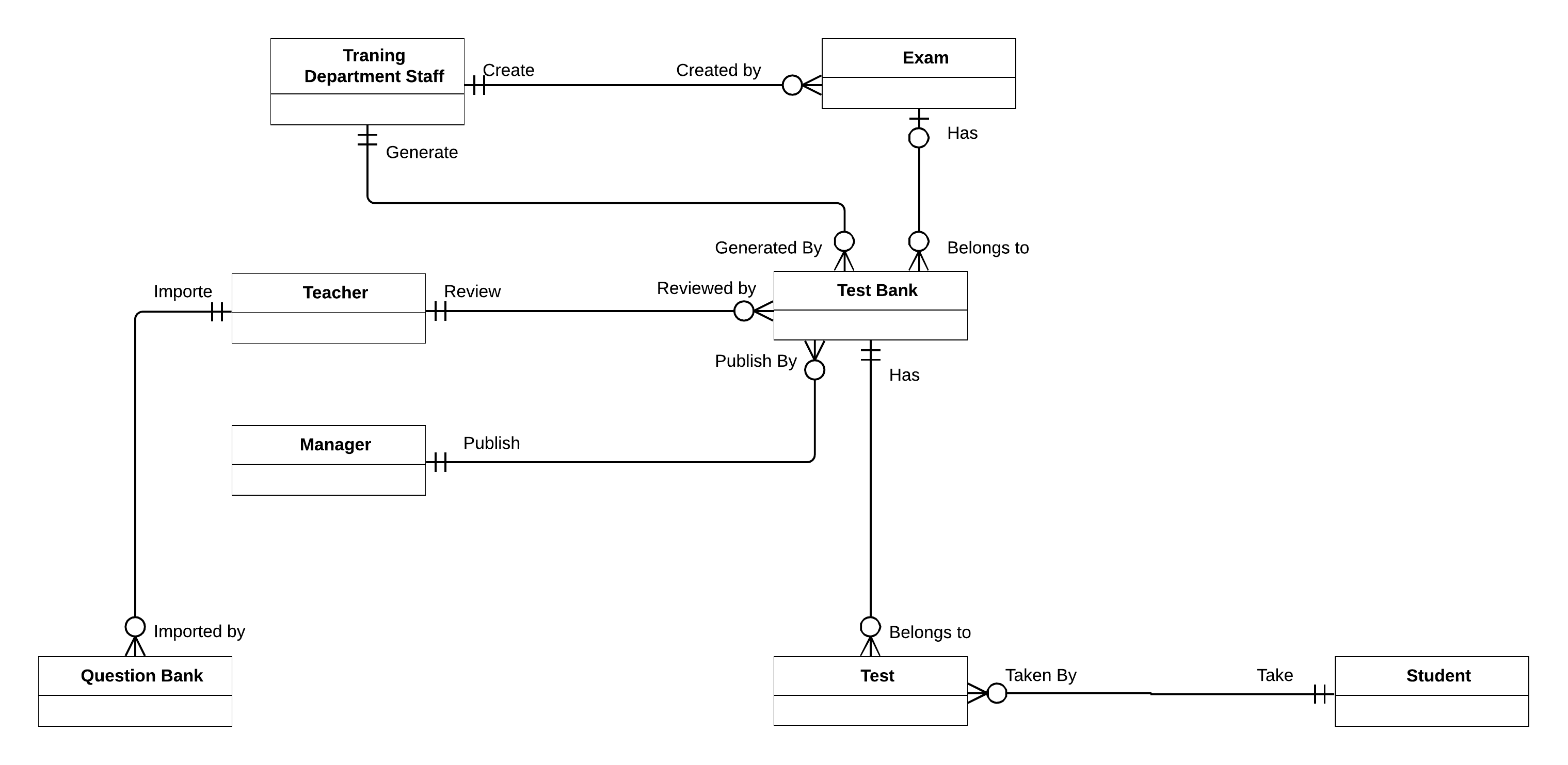


Figure 32 Conceptual Diagram

|  |  |
| --- | --- |
| **Entity Name** | **Description** |
| Training Department Staff | The Staff who generate exam, Test exam, semester. |
| Exam | Created by Training department Staff for every end of semester |
| TestBank | Contain test exam information |
| Teacher | The people who import question and approve test exam |
| Manager | The people who publish test exam |
| QuestionBank | Contain questions information |
| Test | Contain student’s test information |
| Student | Contain Student’s information |

Table 39 CONCEPTUAL DIAGRAM EXPLAINATION

# Software Design Description

## Design Overview

• This document describes the technical and user interface. It includes the architectural design, the detailed design of common functions and business functions and the design of the database model.

• The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

• The detailed design describes a static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

• The database design describes the relationships between entities and details of each entity.

**Document overview:  
Section 2**: gives an overall description of the system architecture design.

**Section 3**: gives component diagrams that describe the connection and integration of the system. **Section 4**: gives the detail design description which includes a class diagram, class explanation, activity diagram and sequence diagram to details the application functions.

**Section 5**: describe screen design.

**Section 6**: describe fully attribute ERD.

**Section 7**: describe algorithms.

## System Architecture Design

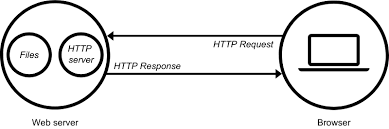


Figure 33 System Architecture Design

### Web Server Architecture Design

Overall architecture of Examination Tool System with functional layers and the collaboration between the system and the external systems is shown in figure 3 below.

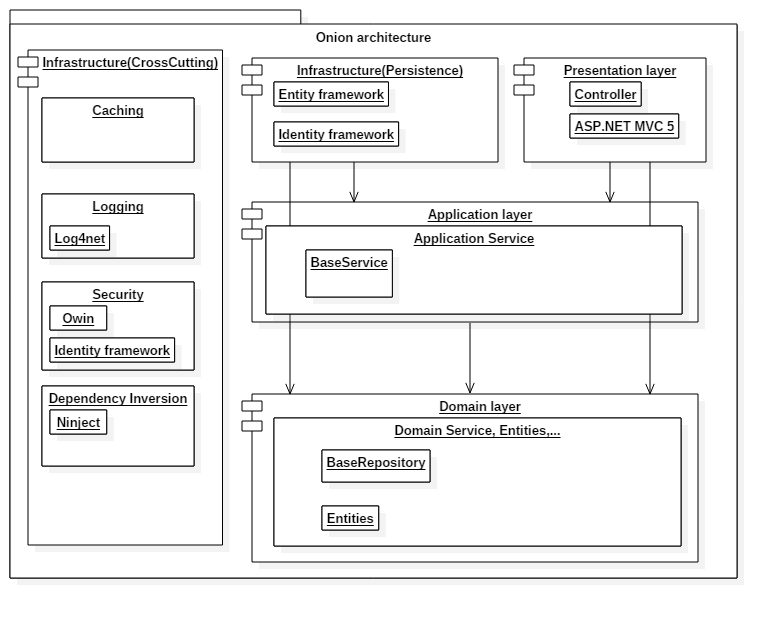


Figure 34 Architecture Overview

|  |  |  |
| --- | --- | --- |
| **No.** | **Layer Name** | **Description** |
| 1 | Presentation layer | Contains UI, handle request |
| 2 | Application layer | Provides services for application |
| 3 | Domain layer | Provides full services, stores entities |
| 4 | Infrastructure layer | Communicate with other framework, work with database |

Table 40 ARCHITECTURE OVERVIEW DESCRIPTION

### Client Architecture Design

In Web Application, the system is developed under MVC architecture. We choose this architecture because of the following advantage:

• The Model-View-Controller pattern highly supports the separation of concerns. This advantage not only increases the testability of the code but it also makes it easier to extend, allowing a fairly easy implementation of new features.

• If the Views respect the single responsibility principle then their role is just to update the Controller for every user event and just display data from the Model, without implementing any business logic. In this case, UI tests should be enough to cover the functionalities of the View.

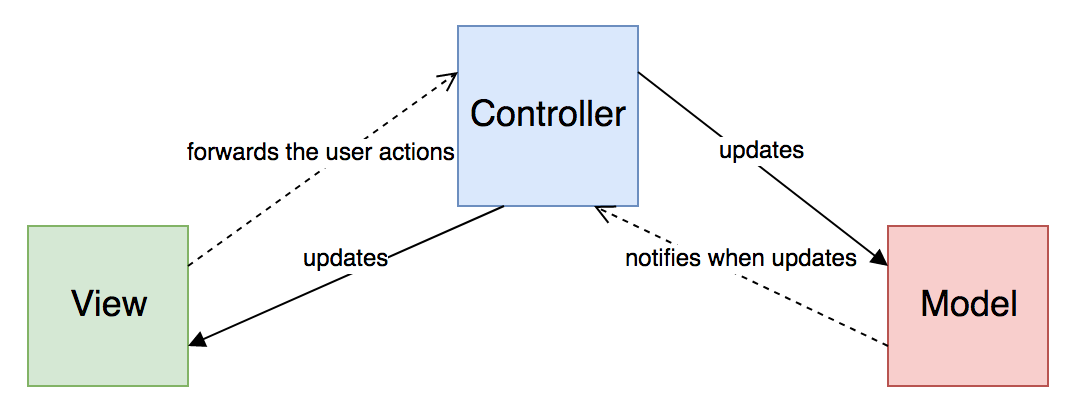


Figure 35 Web Admin MVC Architecture

(Reference: https://medium.com/swlh/ios-design-patterns-a9bd07818129)

* ***Model*:** which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.
* ***View*:** which is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)
* ***Controller*:** which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

## Component Diagram

Đây là component Diagram

## Detailed Description

### Class Diagram

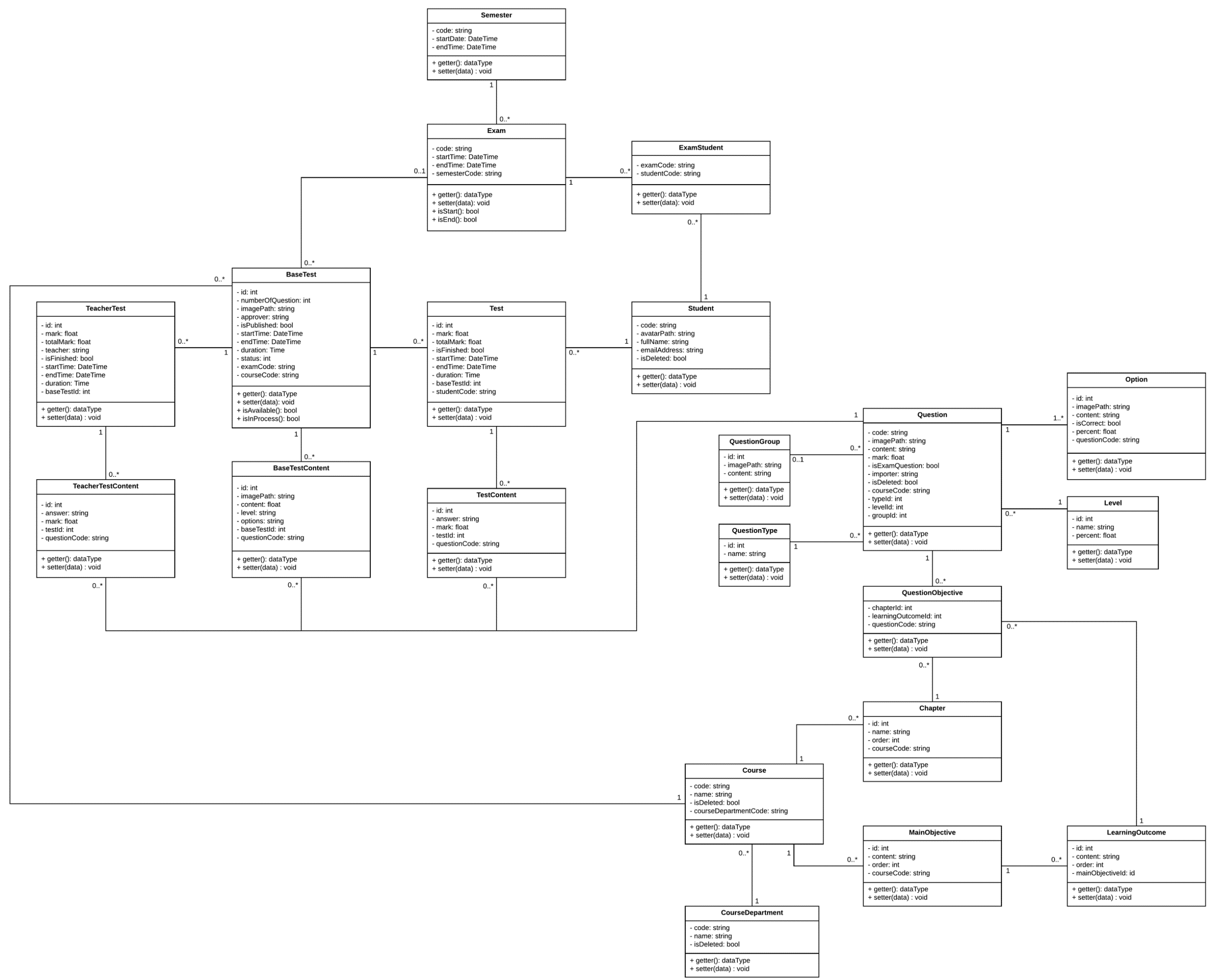


Figure 36 Class diagram

|  |  |  |
| --- | --- | --- |
| **Entity Data dictionary: describe all content of all entities** | | |
| **Class Name** | **Mapping column with conceptual diagram** | **Description** |
| BaseTest | Test | Contain the test exam information |
| Semester | N/A | Contain the Semester information |
| Exam | Exam | Contain the exam information |
| ExamStudent | N/A | Contain relationship between Student and Exam, this describe the accessible to the test exam of student |
| Student | Student | Contain the Student’s information |
| Test | N/A | Contain student’s Test exam information |
| TeacherTest | N/A | Contain teacher’s test exam information (because teacher need to take exam to approve test exam) |
| TeacherTestContent | N/A | Contain answer of each teacher’s question |
| BaseTestContent | N/A | Contain test exam’s question information |
| TestContent | N/A | Contain Student’s answer information |
| Question | QuestionBank | Contain question information (question bank) |
| QuestionGroup | N/A | Contain some group question like reading question which contain some question in reading question |
| Option | N/A | Contain option can choice of each question |
| QuestionType | N/A | Type of question information (ex: single choice, multiple choice) |
| Level | N/A | Contain some level of question (hard, easy …) |
| QuestionObjective | N/A | Question corresponding with LO |
| Chapter | N/A | Contain chapter information |
| Course | N/A | Contain Course information |
| MainObjective | N/A | Contain Purpose of course information |
| LearningOutcome | N/A | Decompose of big Main Objective |
| CourseDepartment | N/A | Contain Course Department information |

Table 41 Class diagram Description

### Class Diagram Explanation

#### Semester

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Code | string | private | Code of Semester (ex: Fall 2018) |
| StartDate | DateTime | private | Start Date of semester |
| EndDate | DateTime | private | End Date of semester |

Table 42 Class Semester Explanation

#### Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Code | string | private | Code of Exam (ex: FinalX) |
| StartDate | DateTime | private | Start Date of Exam |
| EndDate | DateTime | private | End Date of Exam |
| SemesterCode | string | private | Semester Of that Exam |
| CourseCode | string | private | Course of that Exam |

Table 43 Class Exam Explanation

#### ExamStudent

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ExamCode | string | private | Code of Exam (ex FinalX) |
| StudentCode | string | private | Code of a Student (ex SE61801) |

Table 44 Class ExamStudent Explanation

#### Student

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Code | string | private | Code of a Student (ex SE61801) |
| AvatarPath | string | private | Relative image path of Student Avatar |
| FullName | string | private | Student’s Full Name |
| EmailAddress | string | private | Student’s Email |
| isDeleted | boolean | private | Flag to know is this student deleted (true = is deleted) |

Table 45 CLASS STUDENT EXPLANATION

#### BaseTest

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Base Test Id |
| NumberOfQuestion | int | private | Number question of this Test Exam |
| ImagePath | string | private | Relative folder storage image of base test. |
| ExamCode | string | private | Exam of that base test |
| CourseCode | string | private | Course of that base test |
| StartTime | DateTime | private | Start time for student take test exam |
| EndTime | DateTime | private | End time for student take test exam |
| Status | int | private | Status of that base test (approved, rejected, Edited…) |
| Approver | string | private | The people who approve quality of that base test |
| IsPublished | boolean | private | Is that base test published for student |

Table 46 CLASS BASE TEST EXPLANATION

#### BaseTestContent

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Question Id of base test |
| ImagePath | string | private | Relative folder storage image of base test content. |
| Content | string | private | Content of that question |
| Level | string | private | Level of question (describe how question hard) |
| BaseTestId | int | private | Base test of that question |
| QuestionCode | string | private | Code of question (ex PRX-L01) |
| Options | string | private | Json string list option can choice in that question |

Table 47 CLASS BASETESTCONTENT EXPLANATION

#### Test

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Id Student’s Test Exam |
| Mark | double | private | Mark’s Student Test Exam |
| TotalMark | double | private | Total mark of that test exam |
| isFinished | boolean | private | Is student done this test exam |
| StudentCode | string | private | Student who take this test exam |
| StartTime | DateTime | private | Time when student take test exam |
| EndTime | DateTime | private | End time when student take test exam |
| Duration | Time | private | Time of student take that test exam |
| BaseTestId | int | private | Base test of that test exam |

Table 48 CLASS TEST EXPLANATION

#### TestContent

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Id Student’s answer |
| TestId | int | private | Test id of this Student’s answer |
| QuestionCode | double | private | Question code of this Student’s answer |
| Answers | string | private | Json string list student answer of this question |
| Mark | double | private | Mark of this answer in this question |

Table 49 CLASS TESTCONTENT EXPLANATION

#### Question

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Code | string | private | Question Code |
| ImagePath | string | private | Relative path storage image of question |
| Content | string | private | Question content |
| LevelId | int | private | Level of question (hard, easy…) |
| groupId | int | private | Mark of this answer in this question |
| IsExamQuestion | boolean | private | Is question already in test exam |
| CourseCode | string | private | Course of this question |
| Mark | double | Private | Mark of this question |
| isDeleted | boolean | Private | Is this question deleted |
| TypeId | int | Private | Type of question (ex: single choice, multiple choice, matching…) |
| importer | string | Private | The teacher who import this question |

Table 50 CLASS QUESTION EXPLANATION

#### Option

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Id of this option |
| ImagePath | string | private | Relative path storage image of option |
| Content | string | private | Option content |
| IsCorrect | boolean | private | Is this option correct |
| QuestionCode | string | private | Question Code |
| Persent | double | Private | Percent will Minus if choice wrong |

Table 51 CLASS OPTION EXPLANATION

#### Level

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Level Id |
| Name | string | private | Level name (ex: hard, easy) |
| Percent | Float | private | How important the question is |

Table 52 CLASS OPTION EXPLANATION

#### QuestionType

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Type Id |
| Name | string | private | Type name (ex: single choice, multiple choice) |

Table 53 CLASS QUESTIONTYPE EXPLANATION

#### QuestionGroup

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Question Group Id |
| ImagePath | string | private | Relative Path of this question group |
| Content | String | private | Content of this question group |

Table 54 CLASS QUESTION GROUP EXPLANATION

#### QuestionObjective

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ChapterId | int | private | Chapter Id |
| LearningOutcomeId | int | private | Learning Outcome Id |
| QuestionCode | string | private | Question |

Table 55 CLASS QUESTIONOBJECTIVE EXPLANATION

#### Chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Chapter Id |
| Name | string | private | Name of chapter (ex: chapter 1) |
| CourseCode | string | private | Code of course |
| Order | int | private | Order of chapter (1, 2…) |

Table 56 CLASS CHAPTER EXPLANATION

#### Course

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Code | string | private | Course Code |
| Name | string | private | Course Name |
| CourseDepartmentCode | string | private | Department of course |
| IsDeleted | boolean | private | Is this course deleted |

Table 57 CLASS COURSE EXPLANATION

#### MainObjective

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Main Objective Id |
| Content | string | private | Main objective information |
| CourseCode | string | private | Course of this main objective |
| Order | int | private | Order of this main objective |

Table 58 CLASS MAINOBJECTIVE EXPLANATION

#### LearningOutcome

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Learning Outcome Id |
| Content | string | private | Learning outcome information |
| CourseCode | string | private | Big main objective |
| Order | int | private | Order of this Learning outcome |

Table 59 CLASS LEARNINGOUTCOME EXPLANATION

#### TeacherTest

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Teacher Test Id |
| Mark | float | private | Mark of this test exam |
| TotalMark | Float | private | Total mark of this test exam |
| Teacher | string | private | Get from FPT Authenticate Service |
| isFinished | boolean | private | Describe: is this test exam finished |
| startTime | DateTime | private | Time teacher start to take test exam |
| endTime | DateTime | private | Time teacher End take test exam |
| Duration | Time | private | Duration teacher take that test exam |
| baseTestId | Int | Private | Base test id |

Table 60 CLASS TeacherTest EXPLANATION

#### TeacherTestContent

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Id | int | private | Teacher Test Content Id |
| answer | string | private | Answer of question |
| Mark | Float | private | Mark of this answer |
| testId | int | private | Test Exam of this answer |
| QuestionCode | boolean | private | Question of this answer |

Table 61 CLASS TeacherTestContent EXPLANATION

### Interaction Diagram

#### Activity Diagram

Đây là Activity Diagram

#### Sequence Diagram

Đây là Activity Diagram

## Interface

…………………….

## Database Design

### Entity Relationship Diagram

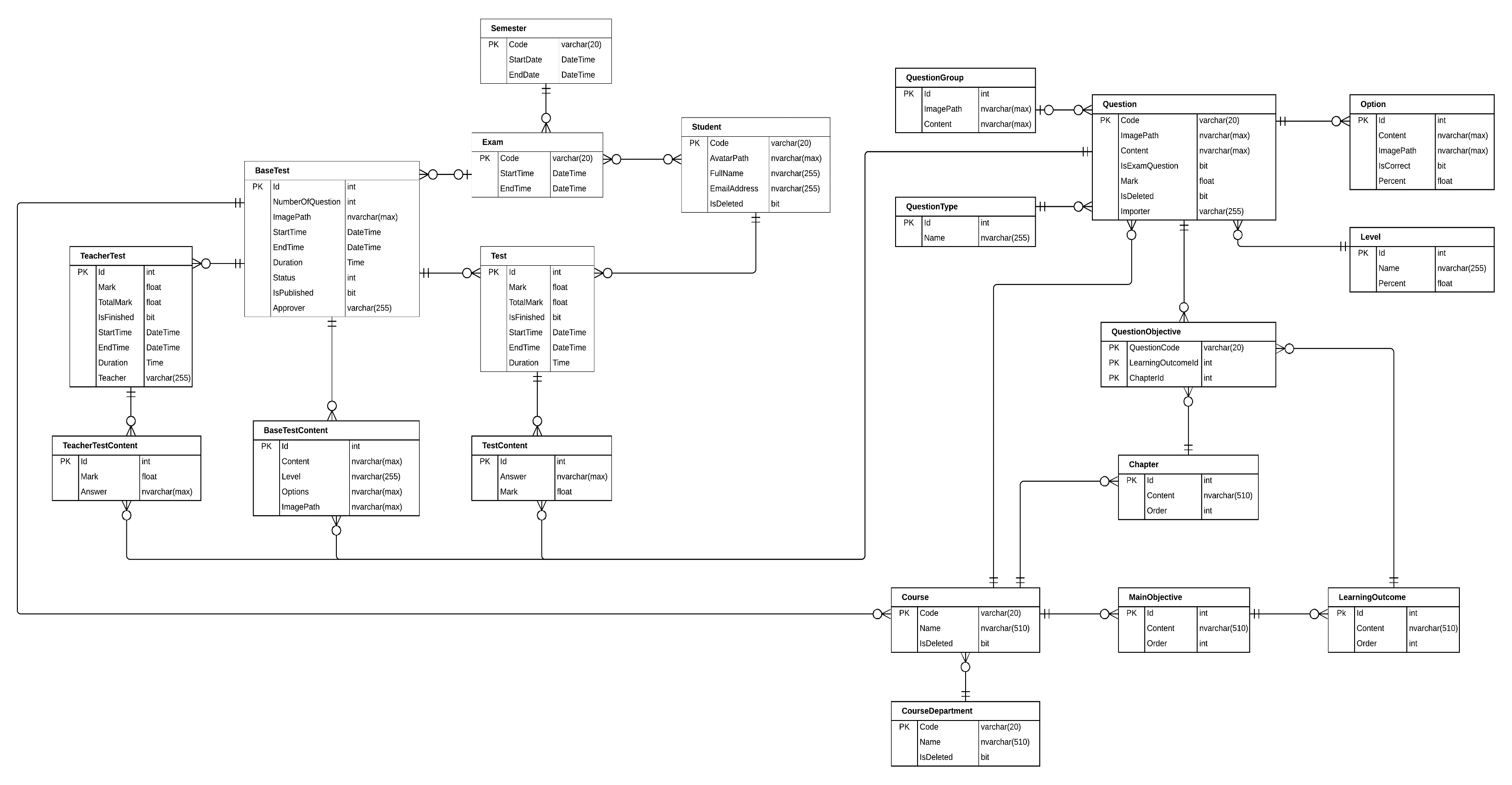


Figure 37 ERD DIAGRAM

### Data Dictionary

|  |  |
| --- | --- |
| **Entity Name** | **Description** |
| BaseTest | Contain the test exam information |
| Semester | Contain the Semester information |
| Exam | Contain the exam information |
| ExamStudent | Contain between Student and Exam, it describe the accessible to the test exam of student |
| Student | Contain the Student information |
| Test | Contain student Test exam information |
| TeacherTest | Contain teacher test exam information (because teacher need to take exam to approve test exam) |
| TeacherTestContent | Contain answer of each teacher’s question |
| BaseTestContent | Contain test exam’s question information |
| TestContent | Contain Student’s answer information |
| Question | Contain question information (question bank) |
| QuestionGroup | Contain some group question like reading question which contain some question in reading question |
| Option | Contain option can choice of each question |
| QuestionType | Type of question information (ex: single choice, multiple choice) |
| Level | Contain some level of question (hard, easy…) |
| QuestionObjective | Question corresponding with LO |
| Chapter | Contain chapter information |
| Course | Contain Course information |
| MainObjective | Contain Purpose of course information |
| LearningOutcome | Decompose of big Main Objective |
| CourseDepartment | Contain Course Department information |

Table 60 ERD Dictionary

### Class Diagram Explanation

#### Semester

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Varchar(20) | Code of Semester (ex: Fall 2018) |
| StartDate | Datetime | Start Date of semester |
| EndDate | Datetime | End Date of semester |

Table 61 Semester table Description

#### Exam

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Varchar(20) | Code of Exam (ex: FinalX) |
| StartDate | Datetime | Start Date of Exam |
| EndDate | Datetime | End Date of Exam |

Table 62 Exam table Description

#### Student

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Varchar(20) | Code of a Student (ex SE61801) |
| AvatarPath | nvarchar(Max) | Relative image path of Student Avatar |
| FullName | nvarchar(255) | Student’s Full Name |
| EmailAddress | nvarchar(255) | Student’s Email |
| isDeleted | bit | Flag to know is this student deleted (true = is deleted) |

Table 63 Student table Description

#### BaseTest

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Base Test Id |
| NumberOfQuestion | int | Number question of this Test Exam |
| ImagePath | nvarchar(max) | Relative folder storage image of base test. |
| StartTime | DateTime | Start time for student take test exam |
| EndTime | DateTime | End time for student take test exam |
| Duration | Time | Duration of test exam |
| Status | int | Status of that base test (approved, rejected, Edited…) |
| Approver | varchar(50) | The people who approve quality of that base test |
| IsPublished | bit | Is that base test published for student |

Table 64 BaseTest table Description

#### BaseTestContent

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Question Id of base test |
| ImagePath | nvarchar(Max) | Relative folder storage image of base test content. |
| Content | nvarchar(max) | Content of that question |
| Level | nvarchar(255) | Level of question (describe how question hard) |
| Options | nvarchar(max) | Json string list option can choice in that question |

Table 65 BaseTestContent table Description

#### Test

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Id Student’s Test Exam |
| Mark | float | Mark’s Student Test Exam |
| TotalMark | float | Total mark of that test exam |
| isFinished | bit | Is student done this test exam |
| StartTime | DateTime | Time when student take test exam |
| EndTime | DateTime | End time when student take test exam |
| Duration | Time | Time of student take that test exam |

Table 66 Test table Description

#### TestContent

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | Id Student’s answer |
| Answer | nvarchar(max) | Json string list student answer of this question |
| Mark | Float | Mark of this answer in this question |

Table 67 TestContent table Description

#### Question

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | nvarchar(20) | Question Code |
| ImagePath | nvarchar(max) | Relative path storage image of question |
| Content | nvarchar(max) | Question content |
| IsExamQuestion | bit | Is question already in test exam |
| Mark | float | Mark of this question |
| isDeleted | bit | Is this question deleted |
| importer | varchar(50) | The people who import this question |

Table 68 Question table Description

#### Option

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Id of this option |
| ImagePath | nvarchar(max) | Relative path storage image of option |
| Content | nvarchar(max) | Option content |
| IsCorrect | bit | Is this option correct |
| Persent | float | Persent will Minus if choice wrong |

Table 69 Option table Description

#### Level

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | Level Id |
| Name | nvarchar(255) | Level name (ex: hard, easy) |
| Percent | float | How important it is |

Table 70 Level table Description

#### QuestionType

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | Type Id |
| Name | nvarchar(255) | Type name (ex: single choice, multiple choice) |

Table 71 QuestionType table Description

#### QuestionGroup

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | QuestionGroup Id |
| ImagePath | nvarchar(max) | Relative Path of this question group |
| Content | nvarchar(max) | Content of this question group |

Table 72 QuestionGroup table Description

#### QuestionObjective

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| ChapterId | int | Chapter Id |
| LearningOutcomeId | int | Learning Outcome Id |
| QuestionCode | varchar(20) | Question |

Table 73 QuestionObjective table Description

#### Chapter

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | Chapter Id |
| Name | nvarchar(255) | Name of chapter (ex: chapter 1) |
| Order | int | Order of chaper (1, 2…) |

Table 74 Chapter table Description

#### Course

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Int | Course Code |
| Name | nvarchar(255) | Course Name |
| IsDeleted | bit | Is this course deleted |

Table 75 Course table Description

#### MainObjective

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | MainObjective Id |
| Content | nvarchar(max) | Main objective information |
| Order | int | Order of this main objective |

Table 76 MainObjective table Description

#### LearningOutcome

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | Int | Learning Outcome Id |
| Content | nvarchar(max) | Learning outcome information |
| Order | int | Order of this Learning outcome |

Table 77 LearningOutcome table Description

#### Course Department

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Varchar(20) | Learning Outcome Id |
| Name | nvarchar(255) | Learning outcome information |
| IsDeleted | bit | Order of this Learning outcome |

Table 80 LearningOutcome table Description

#### TeacherTest

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Teacher Test Id |
| Mark | float | Mark of this test exam of teacher |
| TotalMark | float | Total mark of test test exam |
| isFinished | bit | Is this teacher test finished |
| StartTime | DateTime | Time teacher start to take the test exam |
| EndTime | DateTime | Time teacher end the test exam |
| Duration | Time | Duration teacher done this test exam |
| Teacher | Varchar(255) | Teacher code get from FPT authentication service |

Table 81 TeacherTest table Description

#### TeacherTestContent

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Id | int | Teacher Test Id |
| Mark | float | Mark of this test exam of teacher |
| Answer | nvarchar(max) | Answer of teacher test exam |

Table 82 TeacherTestContent table Description

## Algorithms

### Calculating Percentage Similarity of 2 Question

#### Problem

- When a teacher imports a question list into the question bank, some questions may be duplicated. This can lead to a waste of server’s storage.

#### Solution

In order to solve the mentioned problem, we decide to apply the Levenshtein distance (thêm citation) algorithm.

Levenshtein distance algorithm is used to identify the differences between the range of 2 sequences. The range between these sequences is the minimum steps to make one sequence become the other one. This algorithm includes 3 changing functions:

- Remove a character

- Add a new character

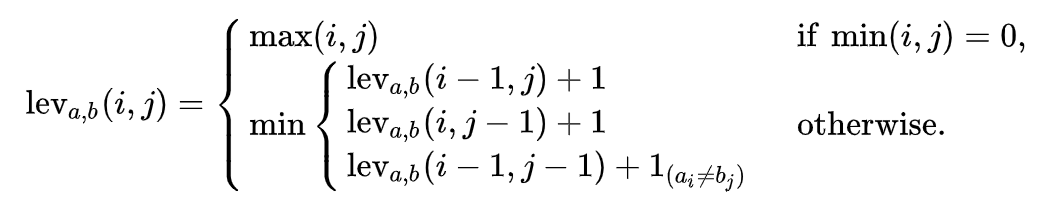
- Replace with another character

Example: To change “kitten” to “sitting”, we have to use at least 3 changing times as following:

1. kitten -> sitten (replace "k" with "s")
2. sitten -> sittin (replace "e" with "i")
3. sittin -> sitting (add "g")

Therefore, the range between "kitten" and sitting" is 3.

The nature of Levenshtein distance algorithm is based on Dynamic Programming. Mathematically, the Levenshtein distance between two strings {\displaystyle a,b}a, b (of length {\displaystyle |a|}|a| and {\displaystyle |b|}|b| respectively) is given by ,{\displaystyle \operatorname {lev} \_{a,b}(|a|,|b|)} where:



With

* {\displaystyle 1\_{(a\_{i}\neq b\_{j})}}  equal to 0 when {\displaystyle a\_{i}=b\_{j}} and equal to 1 otherwise
* {\displaystyle \operatorname {lev} \_{a,b}(i,j)}  is the distance between the first {\displaystyle i} characters of {\displaystyle a} and the first {\displaystyle j} characters of {\displaystyle b}

Note that the first element in the minimum corresponds to deletion (from {\displaystyle a} to {\displaystyle b}), the second to insertion and the third to match or mismatch, depending on whether the respective symbols are the same.

Here is a straightforward pseudocode for a function called LevenshteinDistance that takes two strings:

* *s* of length *m*
* *t* of length *n*

and returns the Levenshtein distance between them:

function LevenshteinDistance(char s[1..m], char t[1..n]):

// for all i and j, d[i,j] will hold the Levenshtein distance between

// the first i characters of s and the first j characters of t

// note that d has (m+1)\*(n+1) values

declare int d[0..m, 0..n]

set each element in d to zero

// source prefixes can be transformed into empty string by

// dropping all characters

for i from 1 to m:

d[i, 0] := i

// target prefixes can be reached from empty source prefix

// by inserting every character

for j from 1 to n:

d[0, j] := j

for j from 1 to n:

for i from 1 to m:

if s[i] = t[j]:

substitutionCost := 0

else:

substitutionCost := 1

d[i, j] := minimum(d[i-1, j] + 1, // deletion

d[i, j-1] + 1, // insertion

d[i-1, j-1] + substitutionCost) // substitution

return d[m, n]

Two examples of the resulting matrix (hovering over a tagged number reveals the operation performed to get that number):

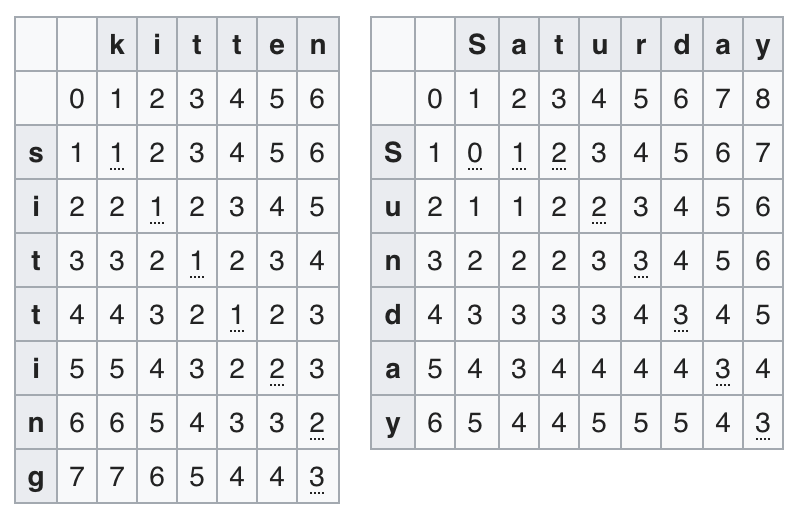


Figure 38 Levenshtein distance example result

#### Complexity

In totally, the complexity of this algorithm is O(n\*m) with n is length of first question and m is length of second question need to compare

# System Implementation & Test

## Introduction

### Overview

This chapter describes the testing and implementation Exam Tool. It includes test plans, test cases, test result and risks estimations and some modification to the previous design phase and system testing to minimize the programming and system error.

### Test Approach

* + Goal: Test all features in the whole Exam Tool system based on the core flow.
  + Method: Interactive system testing.

Testing base on functionality of the software, customer requirements, and implementations. Testers will input data and watch result on the screen. Moreover, we deployment the tool in school’s system, students have to do quiz test in software.

## Database Relationship Diagram

### Physical Diagram:

|  |  |
| --- | --- |
|  | |
| **Table Name** | **Description** |
| BaseTest | Contain the test exam information |
| Semester | Contain the Semester information |
| Exam | Contain the exam information |
| ExamStudent | Contain between Student and Exam, this describe the accessible to the test exam of student |
| Student | Contain the Student information |
| Test | Contain student Test exam information |
| TeacherTest | Contain teacher test exam information (because teacher need to take exam to approve test exam) |
| TeacherTestContent | Contain answer of each teacher’s question |
| BaseTestContent | Contain test exam’s question information |
| TestContent | Contain Student’s answer information |
| Question | Contain question information (question bank) |
| QuestionGroup | Contain some group question like reading question which contain some question in reading question |
| Option | Contain option can choice of each question |
| QuestionType | Type of question information (ex: single choice, multiple choice) |
| Level | Contain some level of question (hard, easy …) |
| QuestionObjective | Question corresponding with LO |
| Chapter | Contain chapter information |
| Course | Contain Course information |
| MainObjective | Contain Purpose of course information |
| LearningOutcome | Decompose of big Main Objective |
| CourseDepartment | Contain Course Department information |
| ImportResult | Result of matching process |
| AuthenticationApp | Store info to Identify Student when using app |
| DefaultProcess | Storage Process must kill |

Table 78 Physical Diagram Description

### Data Dictionary:

#### Semester

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Code | Code of Semester (ex: Fall 2018) |
| StartDate | Start Date of semester |
| EndDate | End Date of semester |

Table 79 SEMESTER TABLE DESCRIPTION

#### Exam

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Code | Code of Exam (ex: FinalX) |
| StartDate | Start Date of Exam |
| EndDate | End Date of Exam |

Table 80 Exam Entity Description

#### Student

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Code | Code of a Student (ex SE61801) |
| AvatarPath | Relative image path of Student Avatar |
| FullName | Student’s Full Name |
| EmailAddress | Student’s Email |
| isDeleted | Flag to know is this student deleted (true = is deleted) |

Table 81 Student Entity Description

#### BaseTest

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Base Test Id |
| NumberOfQuestion | Number question of this Test Exam |
| ImagePath | Relative folder storage image of base test. |
| StartTime | Start time for student take test exam |
| EndTime | End time for student take test exam |
| Duration | Duration of test exam |
| Status | Status of that base test (approved, rejected, Edited…) |
| Approver | The people who approve quality of that base test |
| IsPublished | Is that base test published for student |

Table 82 BaseTest Entity Description

#### BaseTestContent

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Question Id of base test |
| ImagePath | Relative folder storage image of base test content. |
| Content | Content of that question |
| Level | Level of question (describe how question hard) |
| Options | Json string list option can choice in that question |

Table 83 BaseTestContent Entity Description

#### Test

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Id Student’s Test Exam |
| Mark | Mark’s Student Test Exam |
| TotalMark | Total mark of that test exam |
| isFinished | Is student done this test exam |
| StartTime | Tiem when student take test exam |
| EndTime | End time when student take test exam |
| Duration | Time of student take that test exam |

Table 84 Test Entity Description

#### TestContent

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Id Student’s answer |
| Answers | Json string list student answer of this question |
| Mark | Mark of this answer in this question |

Table 85 TestContent Entity Description

#### Question

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Code | Question Code |
| ImagePath | Relative path storage image of question |
| Content | Question content |
| IsExamQuestion | Is question already in test exam |
| Mark | Mark of this question |
| isDeleted | Is this question deleted |
| importer | The people who import this question |

Table 86 Question Entity Description

#### Option

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Id of this option |
| ImagePath | Relative path storage image of option |
| Content | Option content |
| IsCorrect | Is this option correct |
| Persent | Persent will Minus if choice wrong |

Table 87 Option Entity Description

#### Level

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Level Id |
| Name | Level name (ex: hard, easy) |
| Percent | How important it is |

Table 88 Level Entity Description

#### QuestionType

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Type Id |
| Name | Type name (ex: single choice, multiple choice) |

Table 89 QuestionType Entity Description

#### QuestionGroup

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | QuestionGroup Id |
| ImagePath | Relative Path of this question group |
| Content | Content of this question group |

Table 90 QuestionGroup Entity Description

#### QuestionObjective

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| ChapterId | Chapter Id |
| LearningOutcomeId | Learning Outcome Id |
| QuestionCode | Question |

Table 91 QuestionObjective Entity Description

#### Chapter

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Chapter Id |
| Name | Name of chapter (ex: chapter 1) |
| Order | Order of chaper (1, 2…) |

Table 92 Chapter Entity Description

#### Course

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Code | Course Code |
| Name | Course Name |
| IsDeleted | Is this course deleted |

Table 93 Course Entity Description

#### MainObjective

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | MainObjective Id |
| Content | Main objective information |
| Order | Order of this main objective |

Table 94 MainObjective Entity Description

#### LearningOutcome

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Learning Outcome Id |
| Content | Learning outcome information |
| Order | Order of this Learning outcome |

Table 95 LearningOutcome Entity Description

#### AuthenticationApp

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Authentication app Id |
| AppId | Identify the app by Id, this Id generate by signalR Id |
| WebId | Id of web when load by the app identify by AppId. Generated by signalR |
| Username | Student Username |

Table 96 AuthenticationApp table Description

#### AuthenticationApp

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Authentication app Id |
| AppId | Identify the app by Id, this Id generate by signalR Id |
| WebId | Id of web when load by the app identify by AppId. Generated by signalR |
| Username | Student Username |

Table 97 LearningOutcome Entity Description

#### DefaultProcess

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Identify Id |
| AppName | Name of the software run on Student computer |
| ProcessName | Process name |
| Path | Absolute location of software |

Table 98 LearningOutcome Entity Description

## Performance Measures

### Parsing Question

When a lecture import a file, client need at least 3 to 5 seconds to parse 100 questions.

### Validate Parsed Questions

When question contain image, client need maximum 10 seconds to check that image is valid or not.

### Matching Questions of a Course

### Random Question to Create Exam Base on Chapters/ Learning Outcome

When random option is selected, systems need maximum 3 seconds to generate 1 exam of a course.

### Get Exam Test from Server

When the waiting time has expired, Exam Tool should finish get exam request at maximum 15 seconds.

### Submit Exam Tests of a Class

When the duration of a test has expired, Exam Tool should finish submit exam at maximum 10 seconds.

## Test Plan

The purpose of this document is to describe the overall test plan for testing the Exam Tool System. It can verify and ensure that the Exam Tool meets its design specification and other requirements from the user.

### Features to be tested

* **Exam Management**

- Import Question

- Matching Question

- Generate Test

- Approve/Reject Test

* **Exam Tool**

- Do Exam Test

- Submit Exam

### Features not to be tested

* **Exam Management**

- Login

- Logout

- Manage Semester

- Manage Exam Code

- Manage Course

- Manage Chapter

- Manage Learning Outcome

- Manage Course

- Share Screen

- Force Submit

- Export Mark to Excel file

- Review Test

* **Exam Tool**

- Prevent other applications

- Remember Student’s answer

- Prevent multiple logged

## System Testing Test Case

### Manage Exam System Test Case

#### Import Question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **IQ\_1** | Import Question of XML course from Gift Format | 1. Questions parsed from file.  2. Submit Questions. | - All questions was parsed successfully  - Client receive success notify. | Class SE1167, SE1168 | From 18/10/2018  To  18/10/2018 | 194/42 | -There are 26 questions had been duplicated.  - There are 12 questions cannot be added into databases  - There are 4 questions had wrong format with XML content. |

Table 99 Import Question Test Result 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **IQ\_2** | Import Question of Java Web course from Xml file | 1. Questions parsed from file.  2. Submit Questions. | - All questions was parsed successfully  - Client receive success notify. | Class SE1276, SE1675 | From 24/10/2018  To  24/10/2018 | 124/0 | N/A |

Table 100 Import question test result 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **IQ\_3** | Import Question of XML course from XML file | 1. Questions parsed from file.  2. Submit Questions. | - All questions was parsed successfully  - Client receive success notify. | Class SE1166, SE1167, SE1168 | From 29/10/2018  To  29/10/2018 | 175/7 | - 7 questions had wrong format with XML content. |

Table 101 Import question test result 3

**5.1.2 Matching Question**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **MQ\_1** | Compare questions import from file with all questions in question bank | 1. Compare content.  2. Verify Questions. | - All questions was detect successful | N/A | From 25/11/2018  To  26/11/2018 | 45/0 | N/A |

Table 102 Matching Question test result

**5.1.3 Generate Exam**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **GE\_1** | Generate Exam base on Chapters | 1. Setting configuration of Exam  2. Generate Exam | - An Exam was created successfully | Class SE1166, SE1167, SE1168, SE1276, SE1275 | From 24/10/2018  To  30/10/2018 | 7/0 | N/A |

Table 103 Generate Test Result

**5.1.4 Approve Exam**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **AP\_1** | Lecturers take an exam, edit/ swap questions and aprrove/reject the test | 1. Take exam  2. Edit question in the test exam content will auto synchronize into question bank  3. Approve/Reject | - Lecturer can take an exam  - Questions in question bank will be updated when teacher edit questions in text exam content. | N/A | From 23/11/2018  To  27/11/2018 | 112/5 | - Lecturer cannot submit their test 2 times.  - Lecturer cannot update question in question bank 3 times |

Table 104 Approve Exam Test result

**5.2 Exam Tool Test Case**

**5.2.1 Run Exam tool**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_1** | Students run the app and login | 1. Run exam tool  2. Login | - Every students’s laptop can run the app without error.  - Login successful | Class SE1167, SE1168 | From 18/10/2018  To  18/10/2018 | 26/4 | - 4 Laptops cannot run the app. |

Table 105 Run exam tool test result 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_2** | Students run the app and login | 1. Run exam tool  2. Login | - Every students’s laptop can run the app without error.  - Login successful | Class SE1176, SE1175 | From 24/10/2018  To  24/10/2018 | 41/4 | - 4 Laptops cannot run the app. |

Table 106 Run exam tool test result 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_3** | Students run the app and login | 1. Run exam tool  2. Login | - Every students’s laptop can run the app without error.  - Login successful | Class SE1167, SE1168, SE1166 | From 29/10/2018  To  29/10/2018 | 46/5 | - 5 Laptops cannot run the app. |

Table 107 Run exam tool test result 3

**5.2.2 Take Exam**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_1** | Students can take exam then submit their test exam. | 1. Take Exam  2. Submit | - Every students can take an exam  - Every students can submit their test exam | Class SE1167, SE1168 | From 18/10/2018  To  18/10/2018 | 28/2 | - 2 students cannot submit exam. |

Table 108 Take Exam test result 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_2** | Students can take exam then submit their test exam. | 1. Take Exam  2. Submit | - Every students can take an exam  - Every students can submit their test exam | Class SE1176, SE1175 | From 24/10/2018  To  24/10/2018 | 42/3 | - 3 students cannot submit exam. |

Table 109 Take Exam test result 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test case procedure** | **Expected output** | **Inter-test Case Dependence** | **From date to date** | **Success/Fail** | **Note** |
| **RE\_3** | Students can take exam then submit their test exam. | 1. Take Exam  2. Submit | - Every students can take an exam  - Every students can submit their test exam | Class SE1167, SE1168, SE1166 | From 29/10/2018  To  29/10/2018 | 47/4 | - 3 students cannot submit exam.  - 1 student lost exam test because of window update. |

Table 110 Take Exam test result 3

## System Implement

### Web Server Application

Overall architecture of Examination Tool System with functional layers and the collaboration between the system and the external systems is shown in figure 3 below.

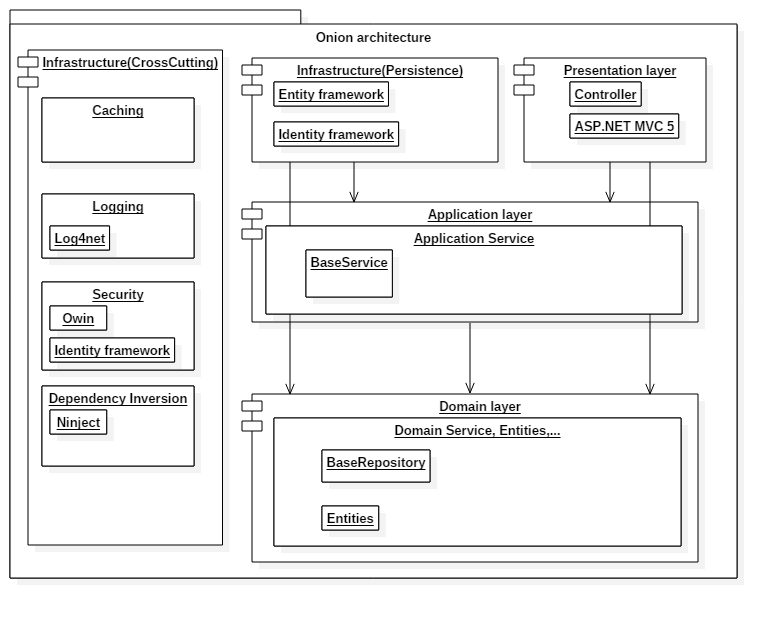


Figure 39 Onion Architecture

|  |  |  |
| --- | --- | --- |
| **No.** | **Layer Name** | **Description** |
| 1 | Presentation layer | Contains UI, handle request |
| 2 | Application layer | Provides services for application |
| 3 | Domain layer | Provides full services, stores entities |
| 4 | Infrastructure layer | Communicate with other framework, work with database |

Figure 40 Onion Architecture description

### Web Client Application

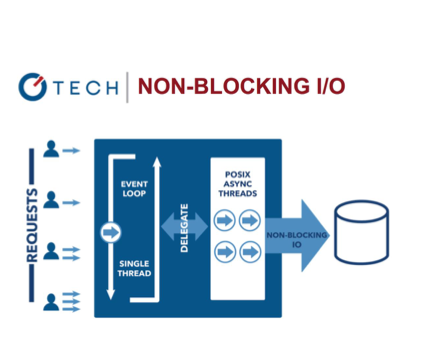
• Angular separate page to many components, each component is reusable, which make the application easy to debugging, maintenance.

• Component: The component controls the display, control View, so you can imagine the Component as a controller in the MVC model.

• Service classes are capable of performing some commonly used functions. Some common services are: logging service, data service, message bus, ...

• Using dependency injection: Allow the creation of class objects with all the additional modules / modules / services.

• The event-driven architecture caters to both the client-side and the server- side that are written in JavaScript and thus the synchronization process is fast and orderly. The event loop through web socket protocol which works on TCP handles the multi-user function and prevents the overhead of HTTP for web development.



In Web client application, we are using Angular under MVC architecture. We choose this Angular because of the following advantage:

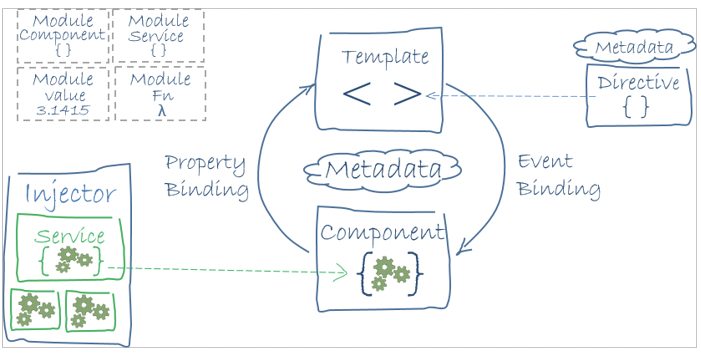


Figure 41 Angular Component Example

Reference: “https://angular.io/guide/architecture”

# Software User’s Manual

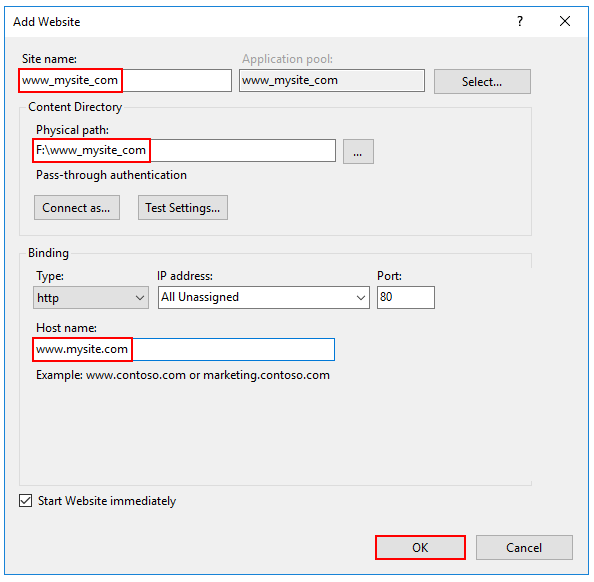
## Installing Guide

1. On the hosting system, create a folder to contain the app's published folders and files. An app's deployment layout is described in the [Directory Structure](https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/directory-structure?view=aspnetcore-2.1) topic.
2. Within the new folder, create a *logs* folder to hold ASP.NET Core Module stdout logs when stdout logging is enabled. If the app is deployed with a *logs* folder in the payload, skip this step. For instructions on how to enable MSBuild to create the *logs* folder automatically when the project is built locally, see the [Directory structure](https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/directory-structure?view=aspnetcore-2.1) topic.

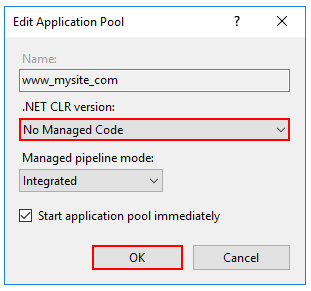
**Important**

Only use the stdout log to troubleshoot app startup failures. Never use stdout logging for routine app logging. There's no limit on log file size or the number of log files created. The app pool must have write access to the location where the logs are written. All of the folders on the path to the log location must exist. For more information on the stdout log, see [**Log creation and redirection**](https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/aspnet-core-module?view=aspnetcore-2.1#log-creation-and-redirection). For information on logging in an ASP.NET Core app, see the [**Logging**](https://docs.microsoft.com/en-us/aspnet/core/fundamentals/logging/index?view=aspnetcore-2.1) topic.

1. In **IIS Manager**, open the server's node in the **Connections** panel. Right-click the **Sites** folder. Select **Add Website** from the contextual menu.
2. Provide a **Site name** and set the **Physical path** to the app's deployment folder. Provide the **Binding**configuration and create the website by selecting **OK**:



1. Under the server's node, select **Application Pools**.
2. Right-click the site's app pool and select **Basic Settings** from the contextual menu.
3. In the **Edit Application Pool** window, set the **.NET CLR version** to **No Managed Code**:



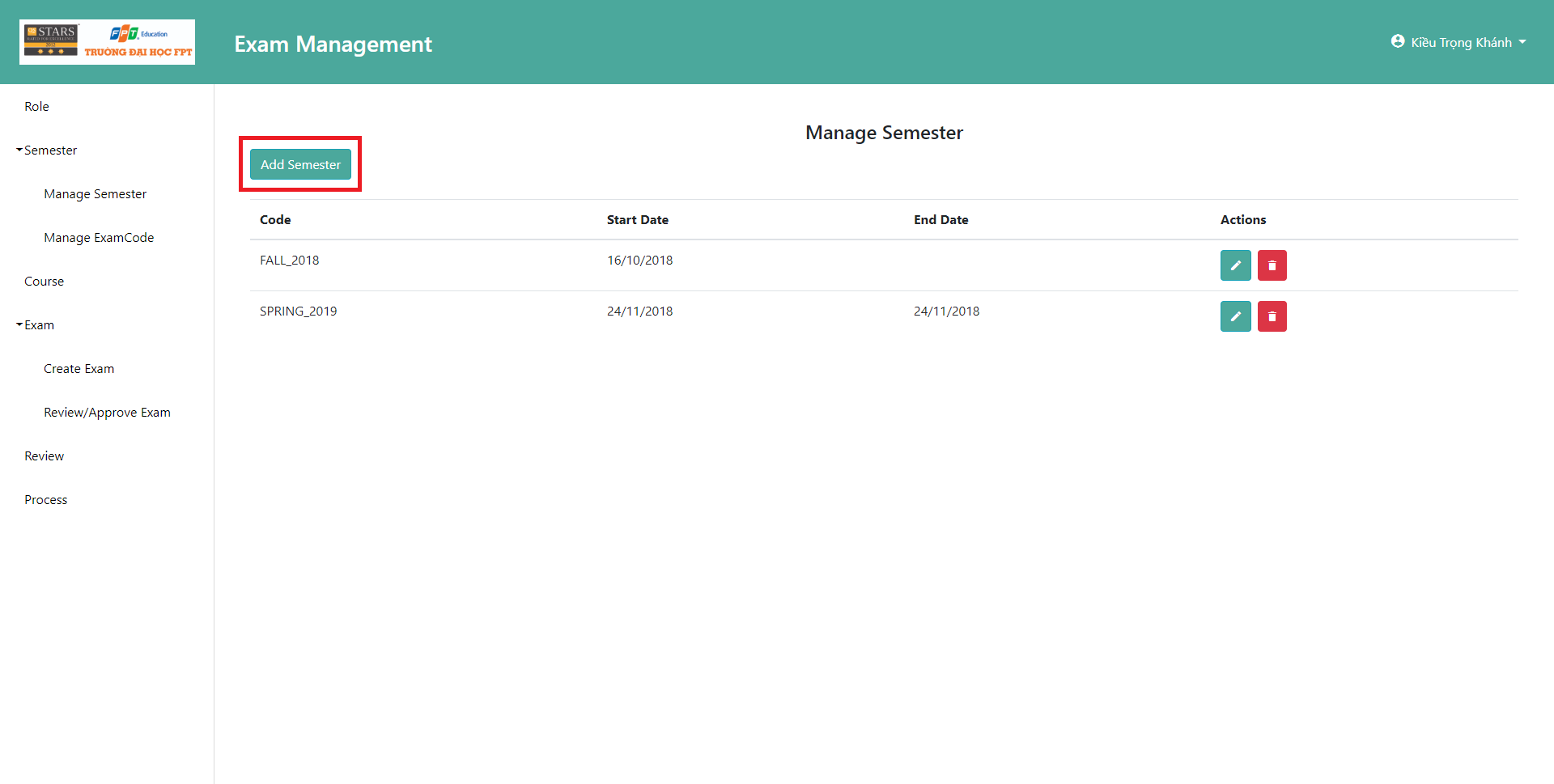
ASP.NET Core runs in a separate process and manages the runtime. ASP.NET Core doesn't rely on loading the desktop CLR. Setting the **.NET CLR version** to **No Managed Code** is optional.

1. Confirm the process model identity has the proper permissions.

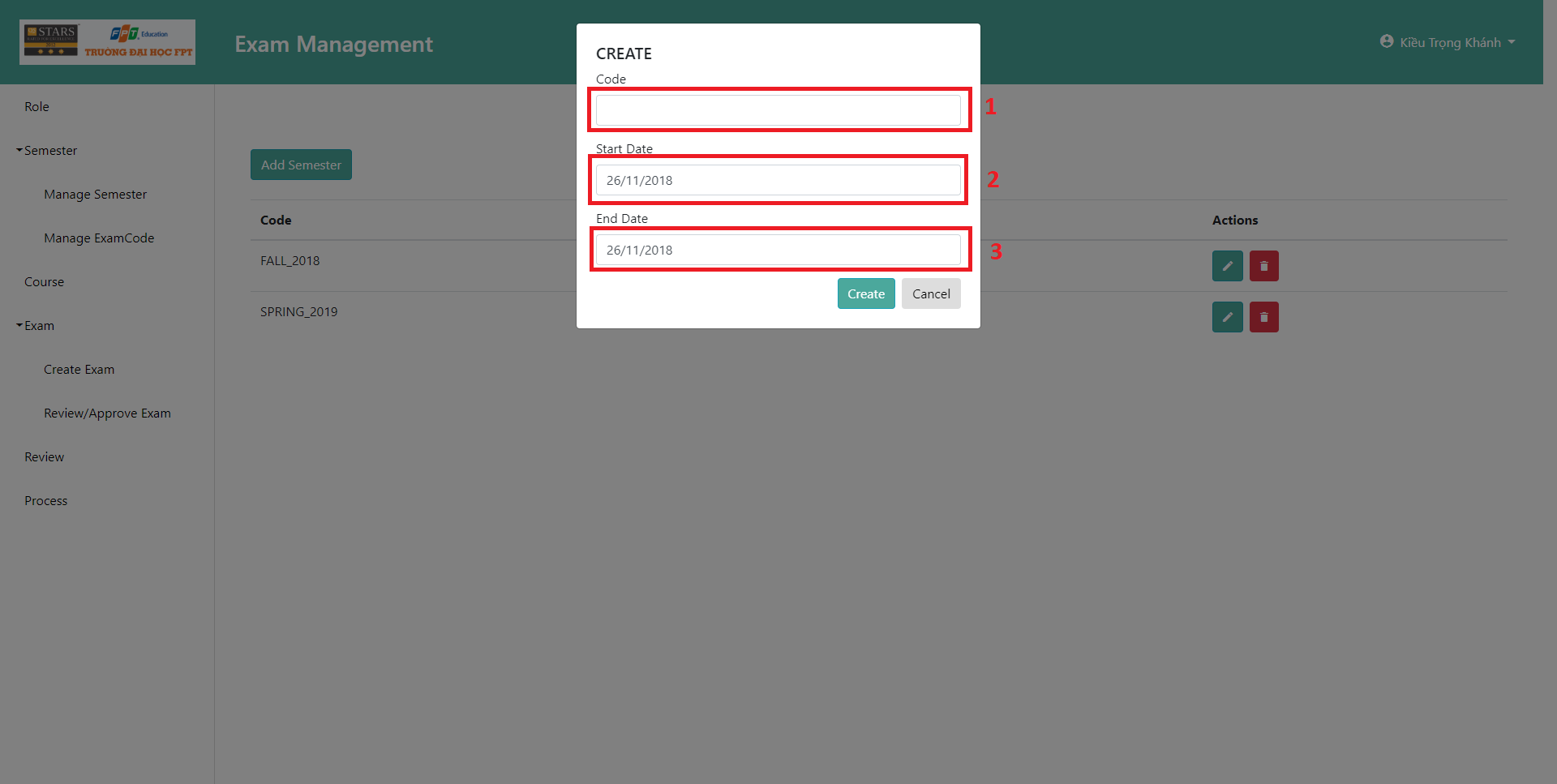
If the default identity of the app pool (**Process Model** > **Identity**) is changed from **ApplicationPoolIdentity** to another identity, verify that the new identity has the required permissions to access the app's folder, database, and other required resources. For example, the app pool requires read and write access to folders where the app reads and writes files.

## User Guide

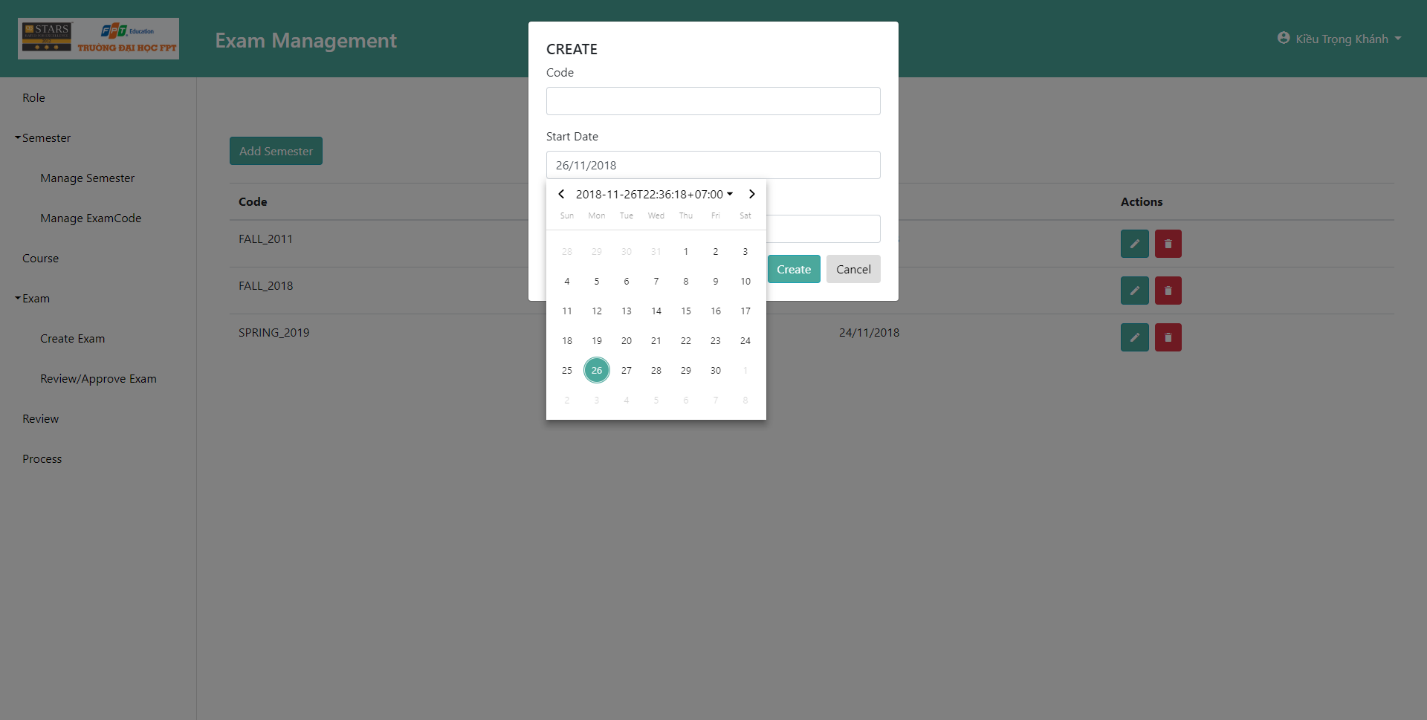
### Admin Create Semester



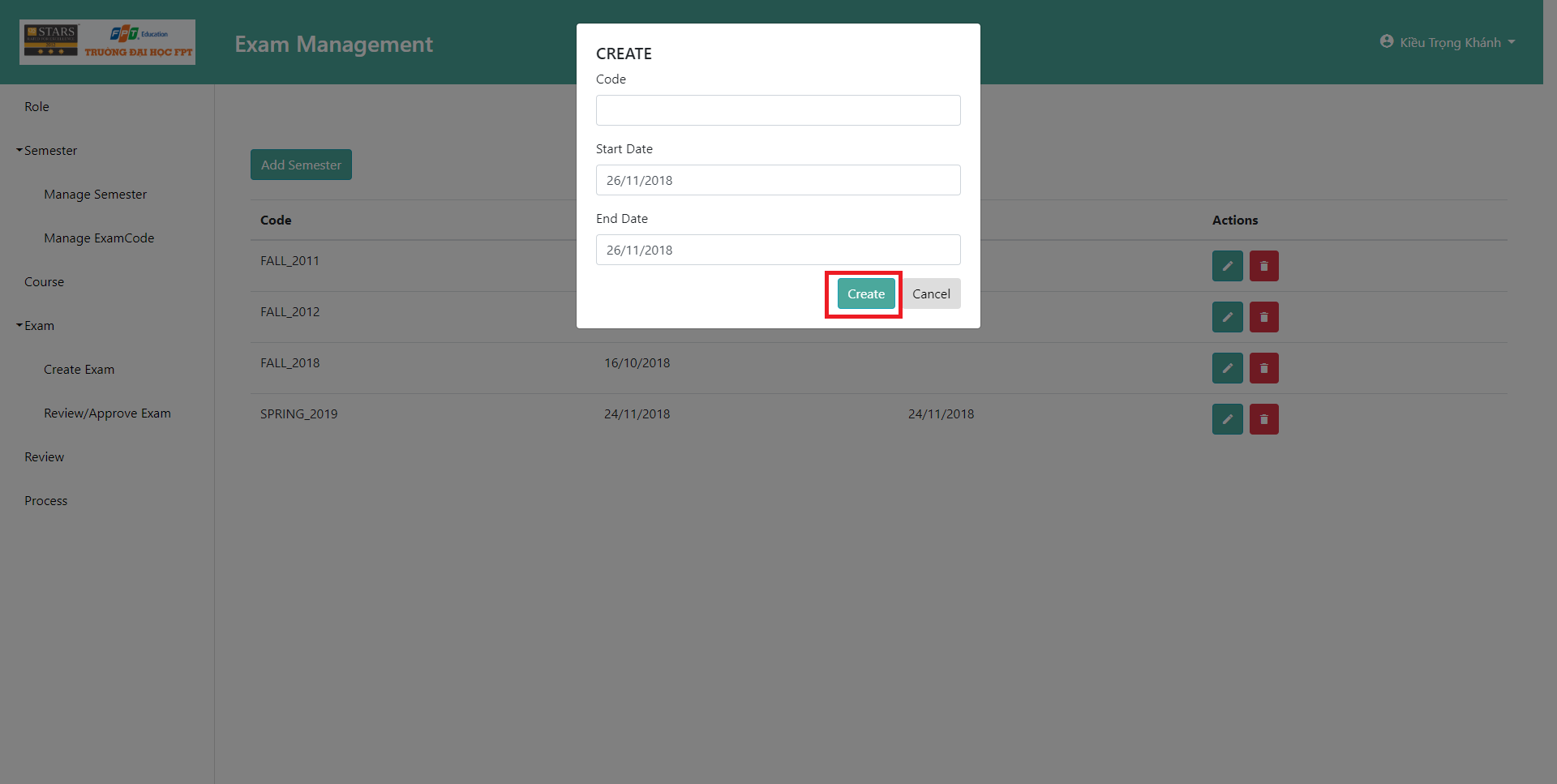
Admin Click the “Add Semester” button to start adding new semester.



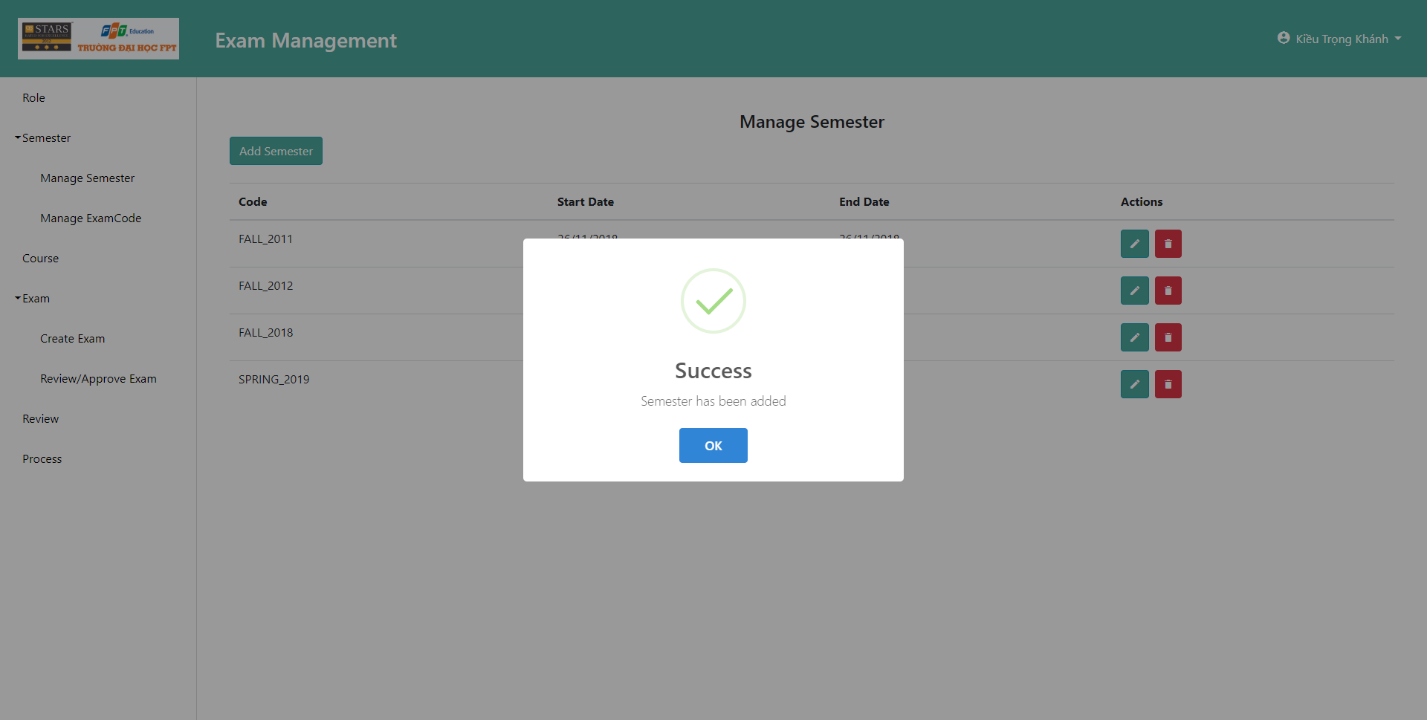
Admin input the Semester Code (1) and the date for start and end of the semester (2) (3)



A Date picker will show up, admin can choose a date from it.

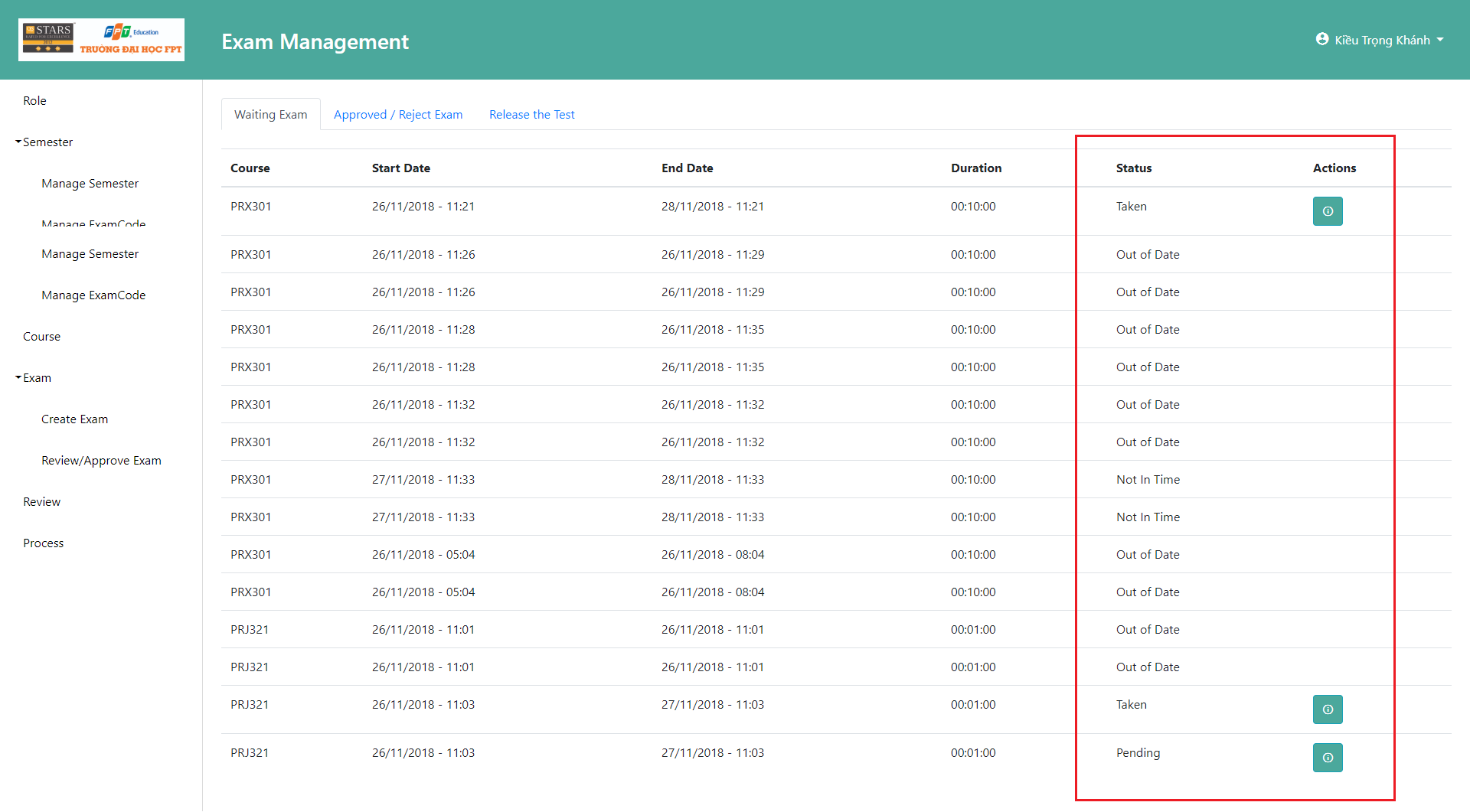


After complete all input fields, admin can click the “Create” button to complete and create a new semester.

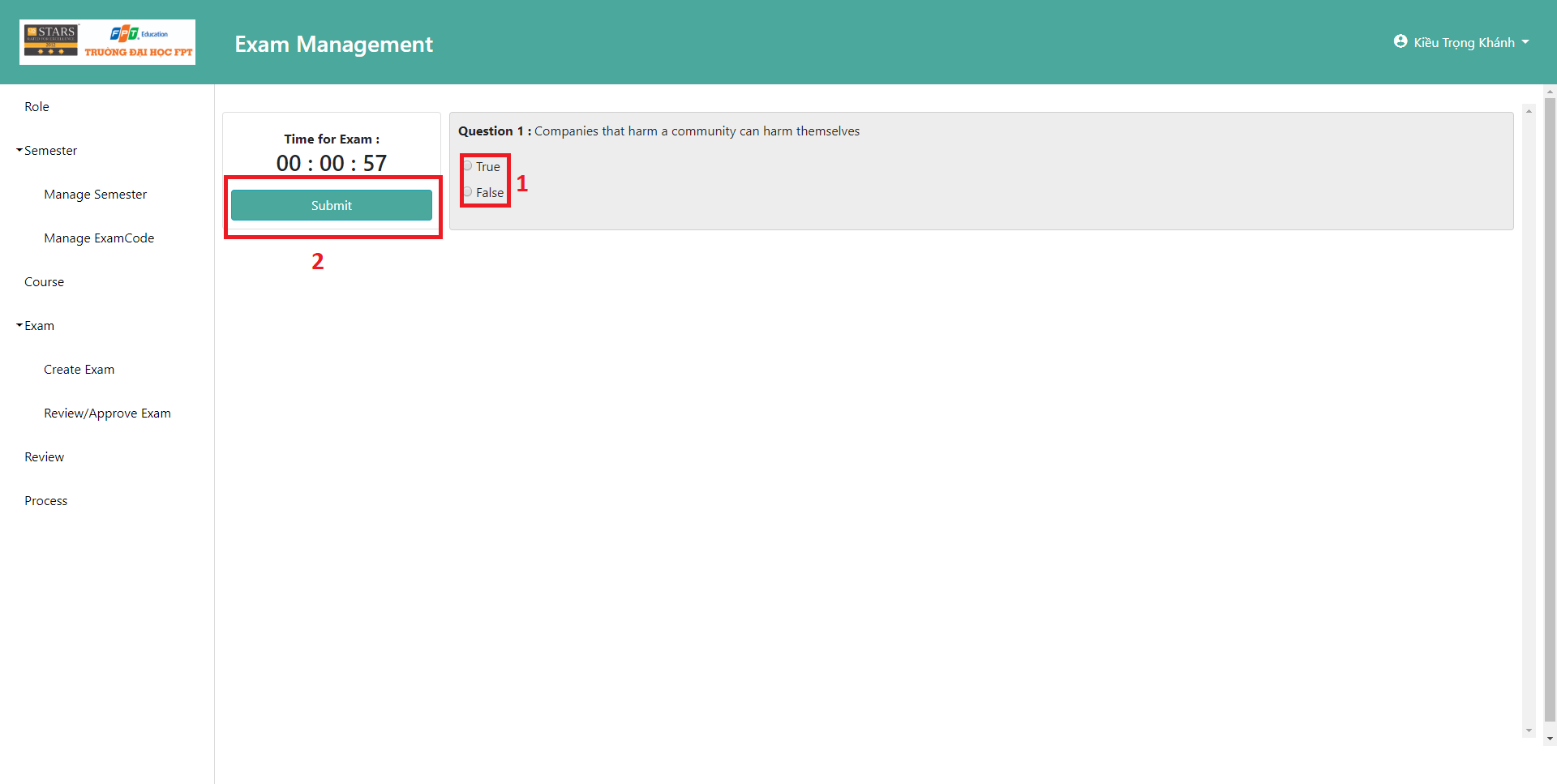


A Popup will indicate that the new semester has been successfully created or not.

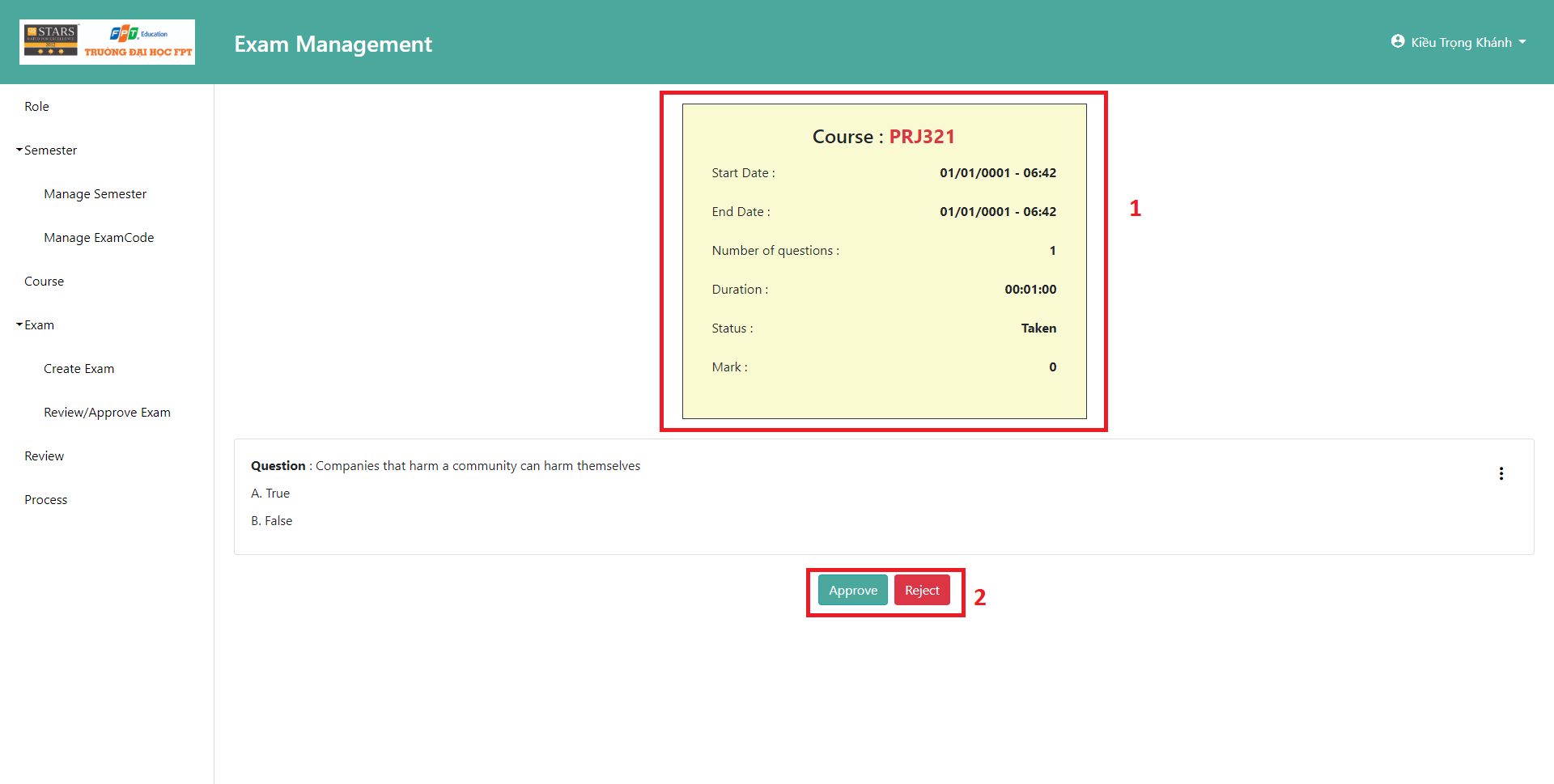
### Teacher approve/reject exam



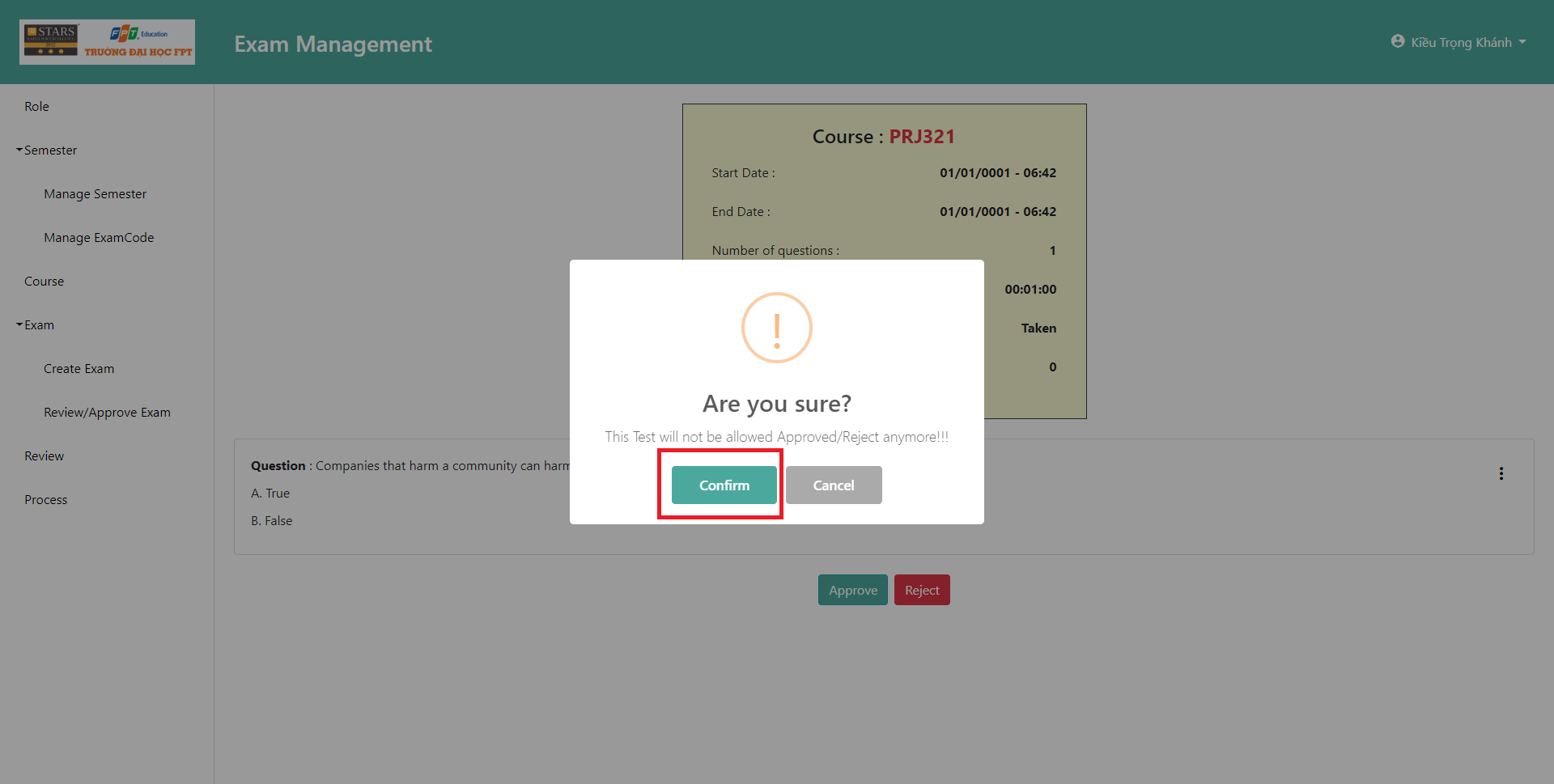
Teacher go to the “Review/Approve Exam” Page and can take action to approve or reject exams. Exams which status is “Pending” or “Taken” can be taken action.



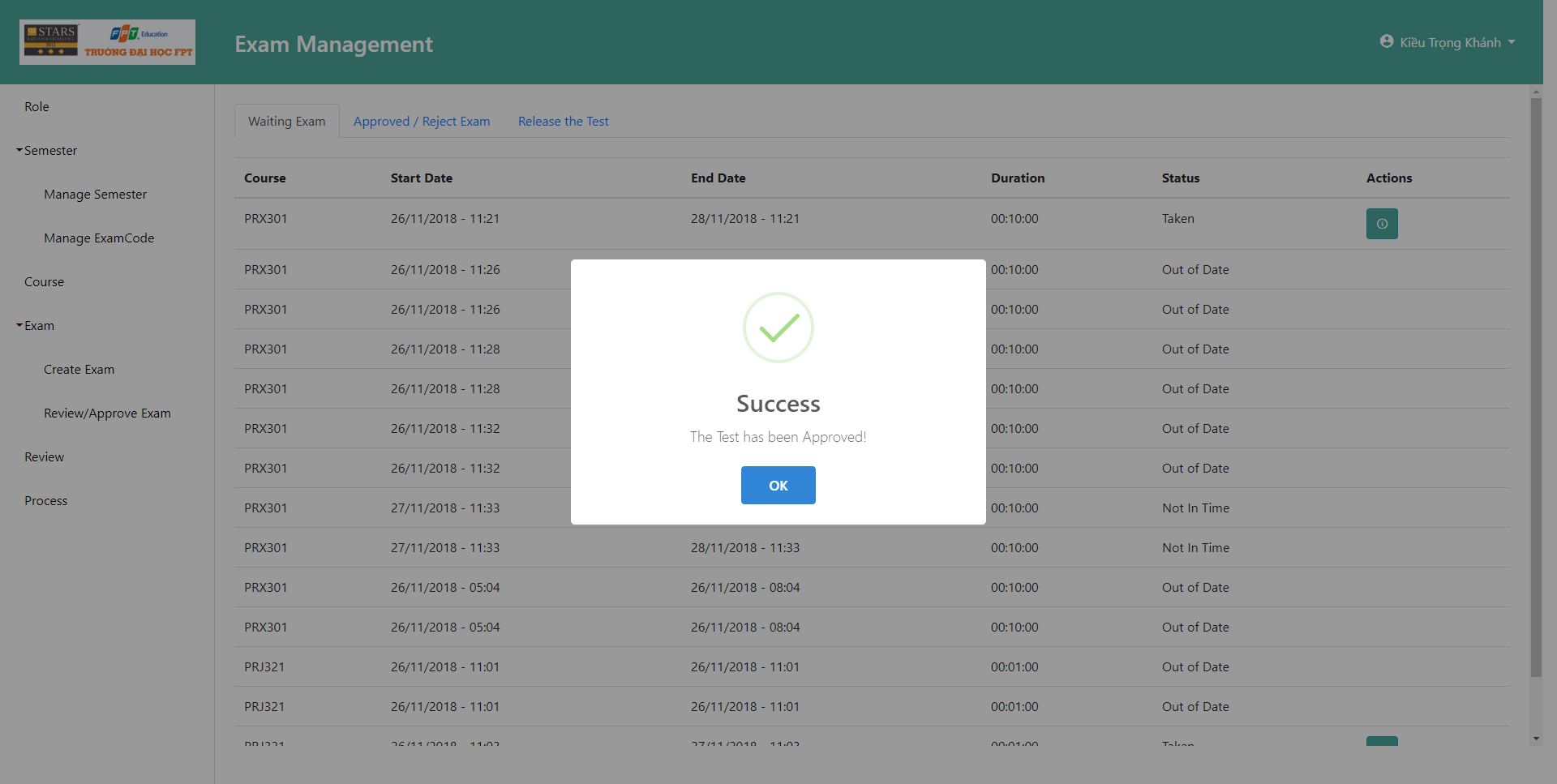
Teacher can either do the test (1) or submit it (2).



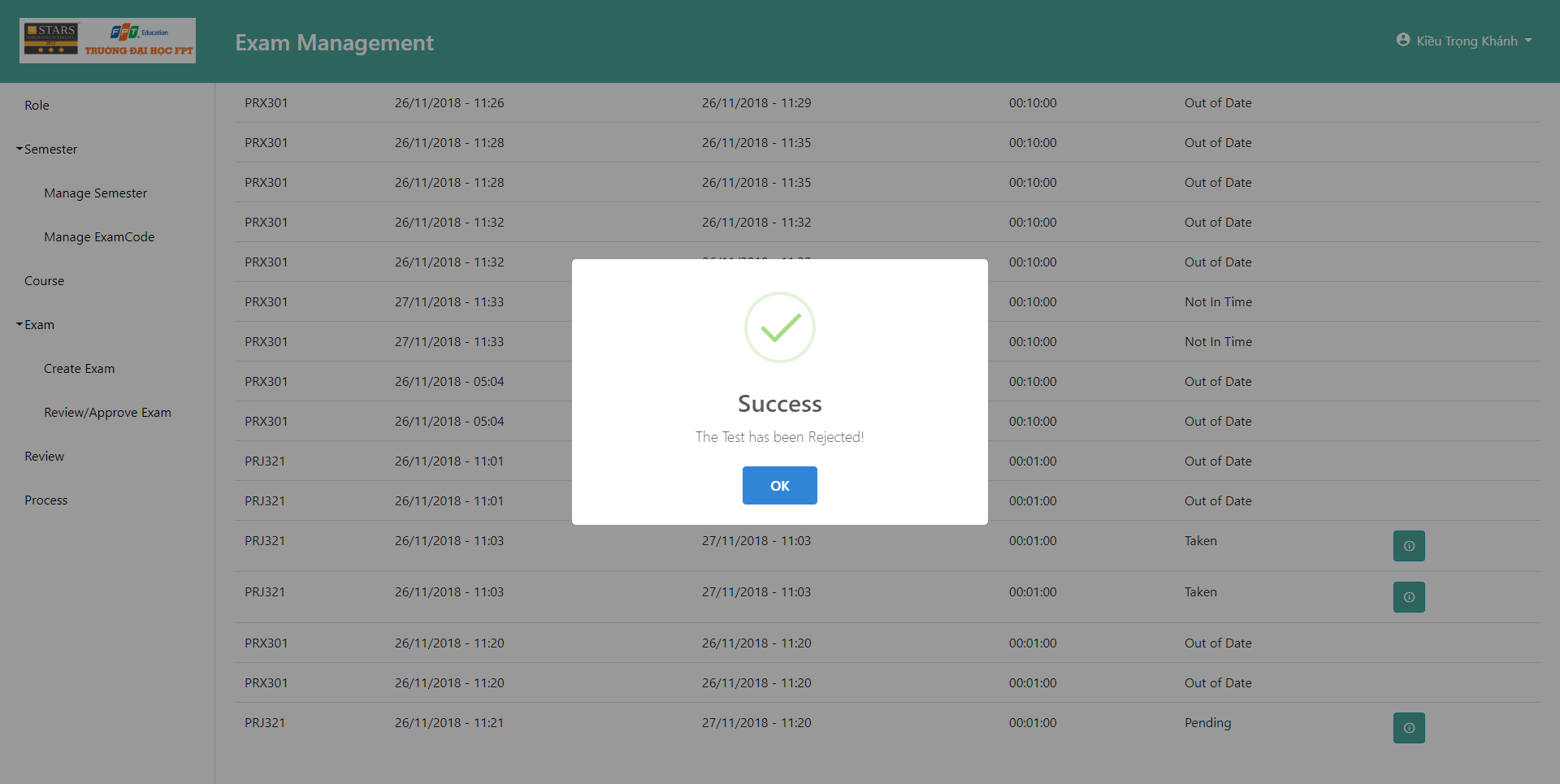
After submitting the test, a summary about the test will be shown (1). Teacher then can click the “Approve” or “Reject” button.



Click confirm to Approve or Reject the Test Exam.

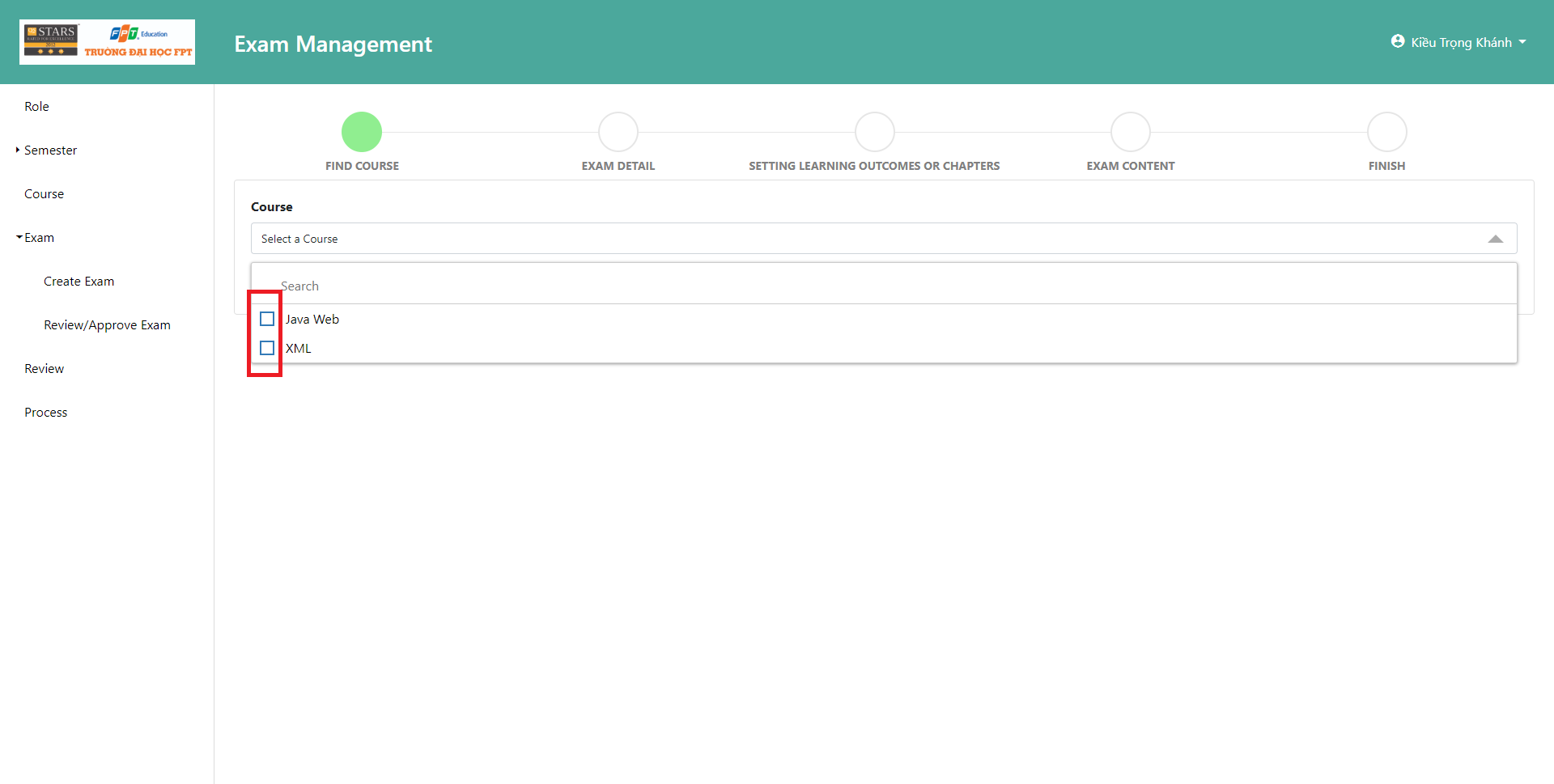


Popup will be shown to indicate that the test exam is approved.



Popup will be shown to indicate that the test exam is rejected.

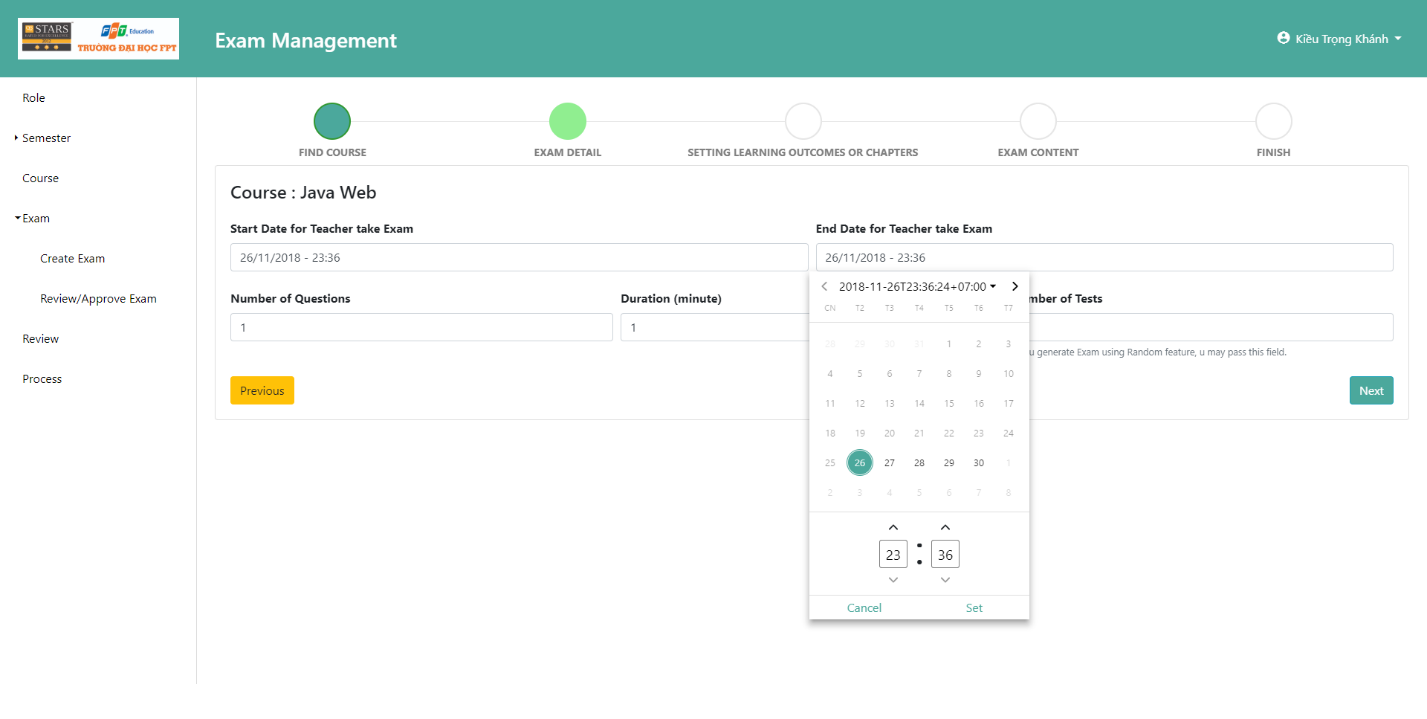
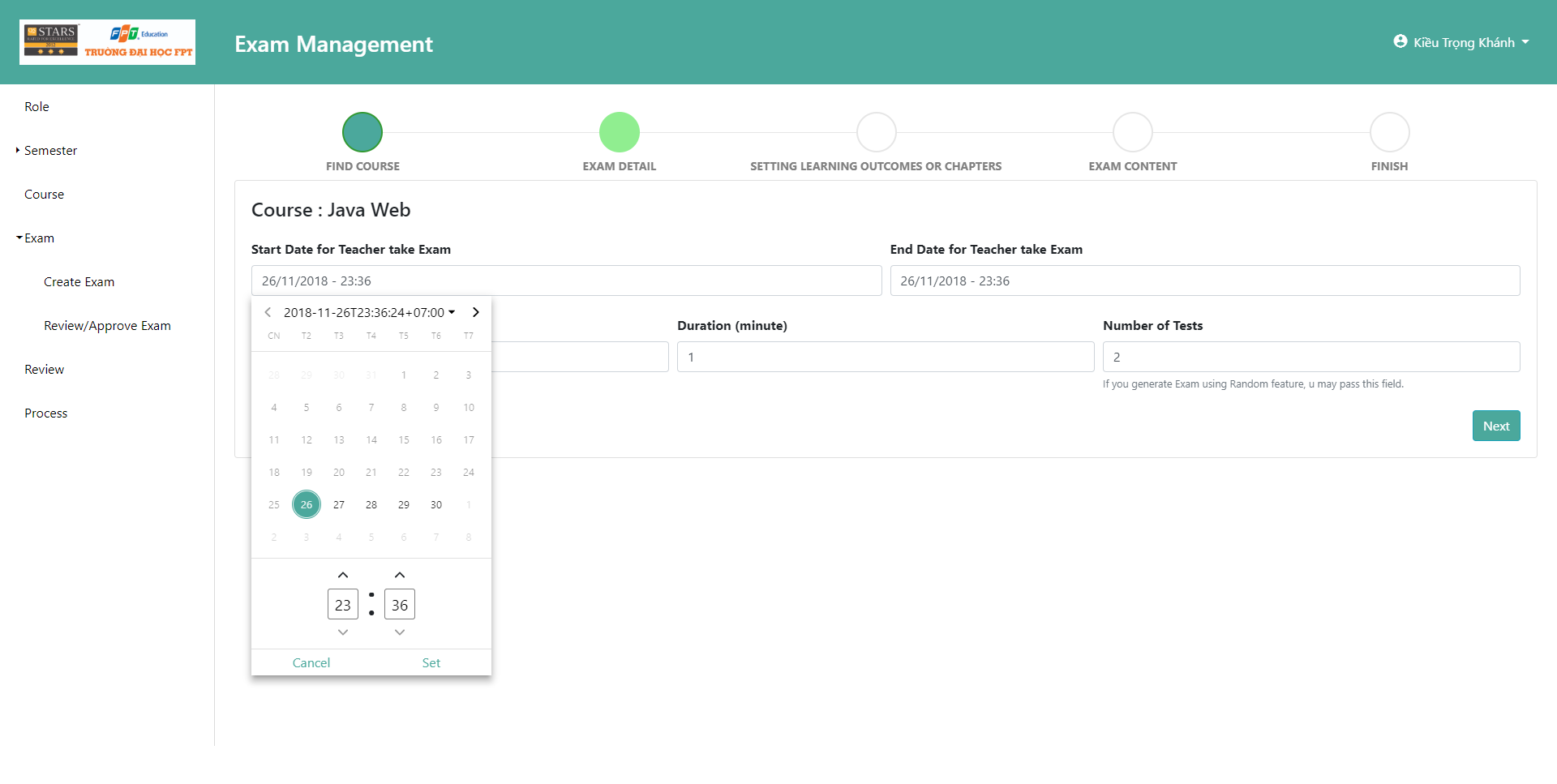
### Teacher Create Exam



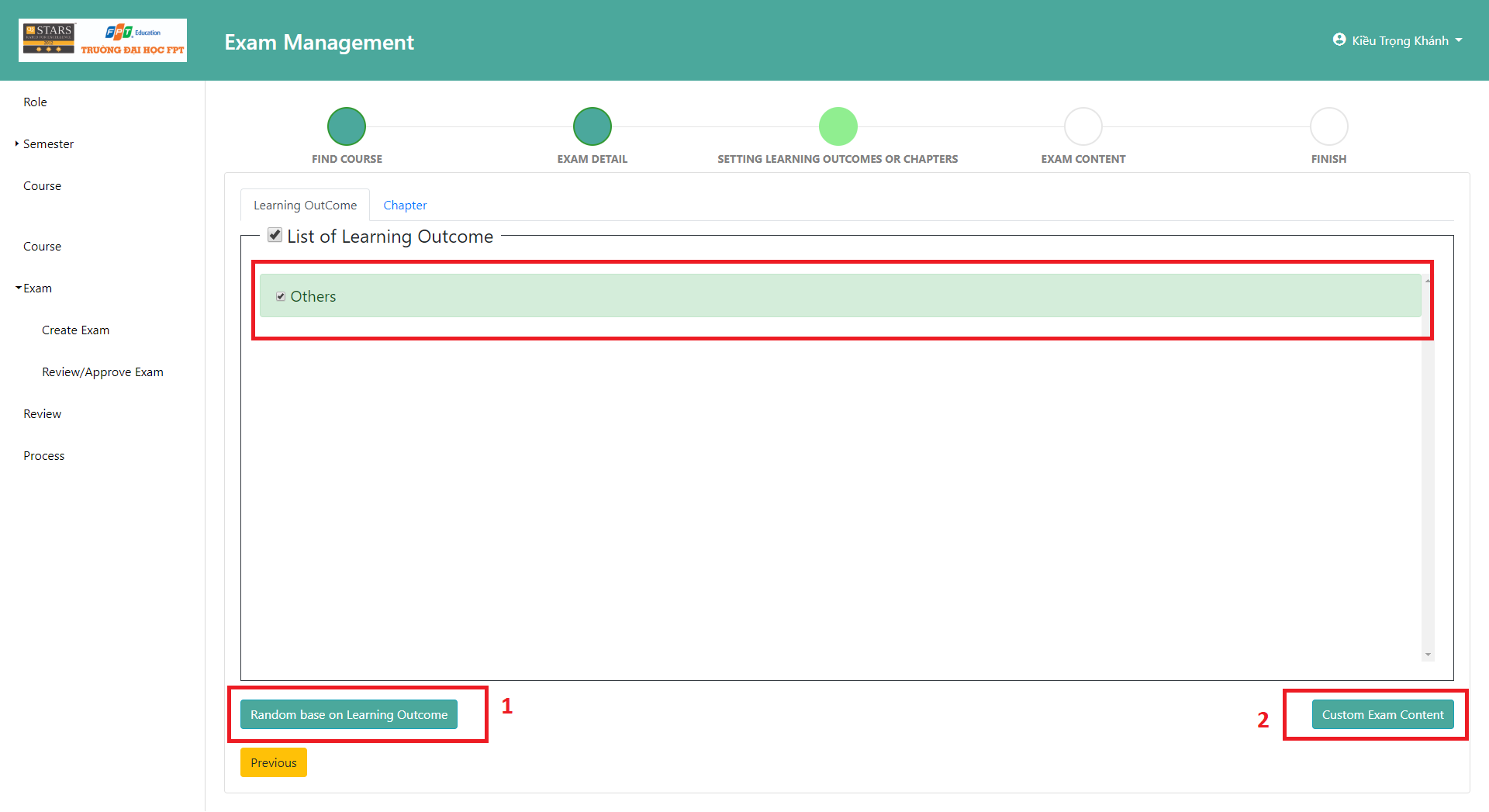
Teacher go to the Create Exam page , a create wizard will appear , first is to choose the course.



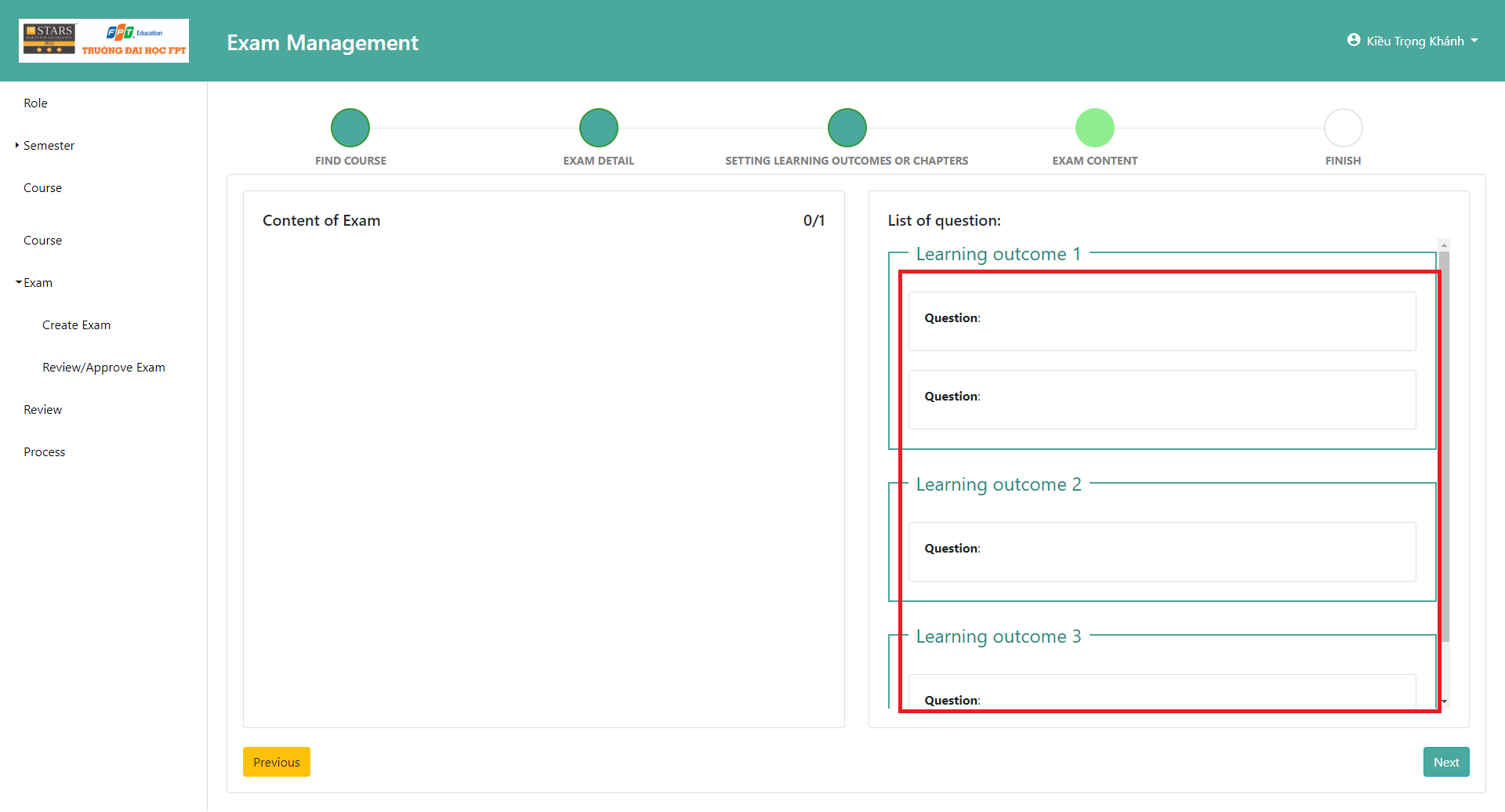
After choosing which course will have the test created, teacher must fill in 4 required fields (1)(2)(3)(4), (5) is left for optional if learning outcome is chosen in the next stage.

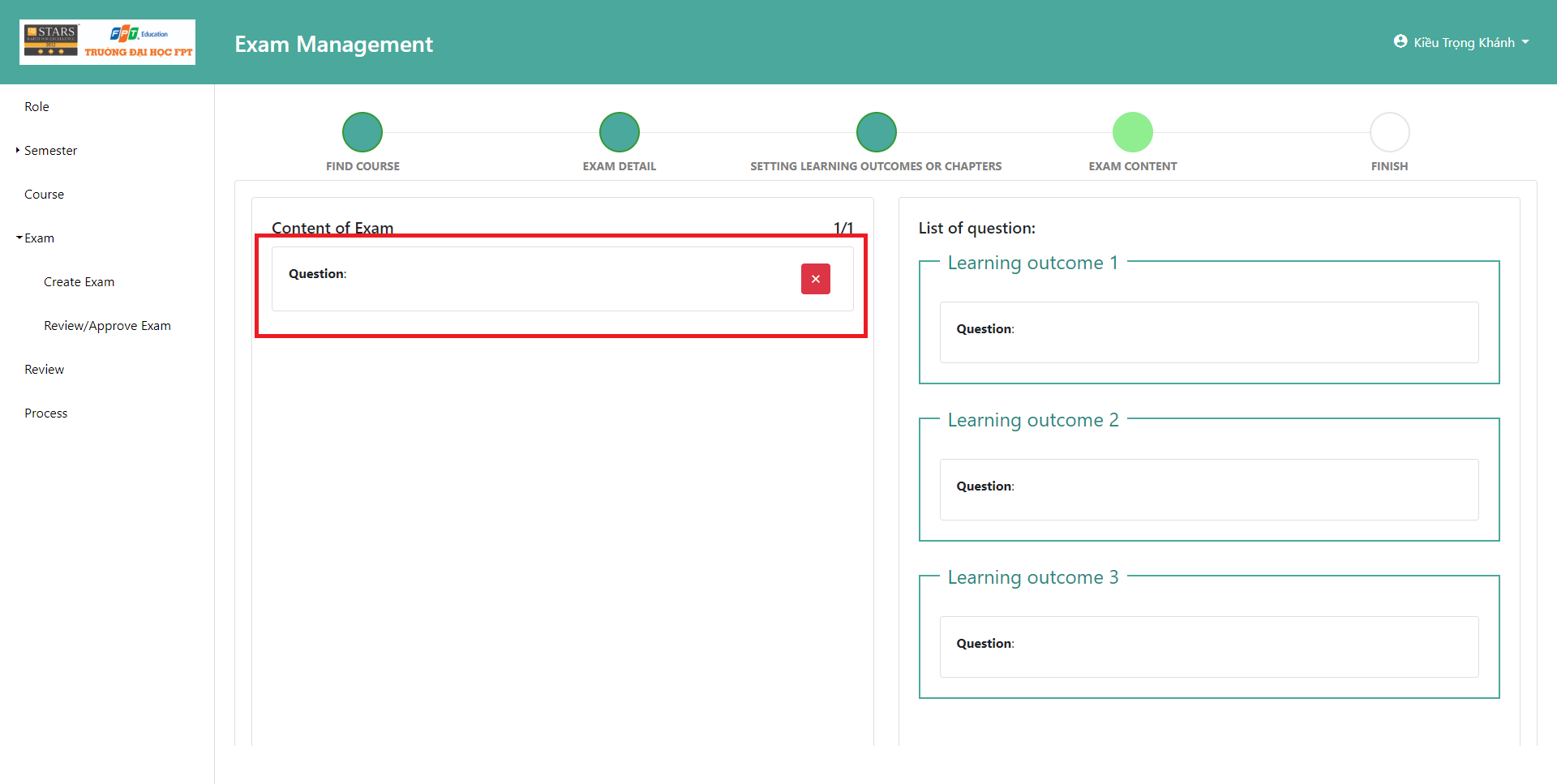


A date time picker will be show up to choose the start and end date time for teacher to take the exam.

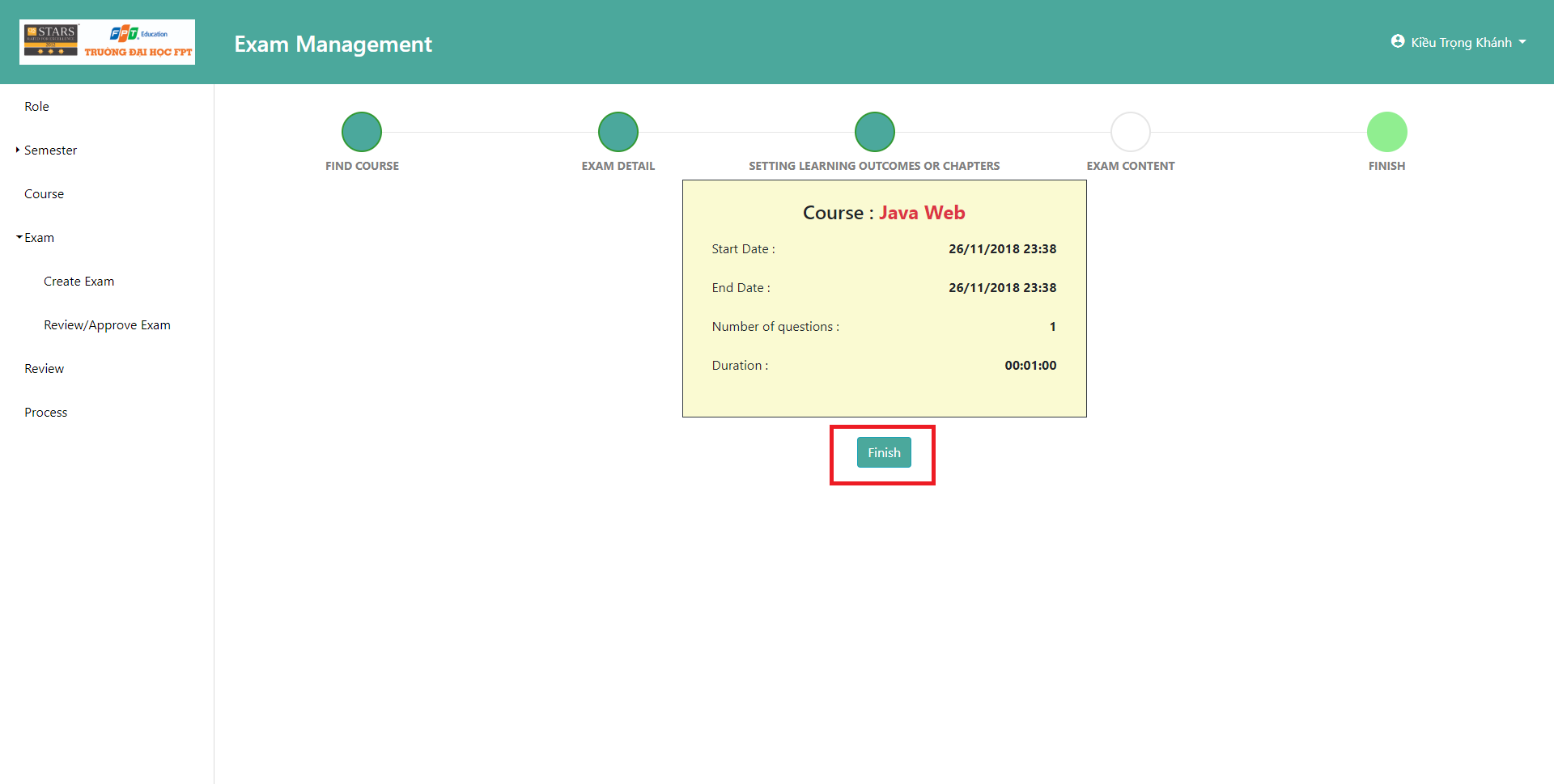


Teacher can choose which Learning Outcome they want in the test exam. They can randomly choose questions base on Learning Outcome by clicking (1) or manually custom the test exam questions by clicking (2).





Questions can be drag and drop from the List to the Content of Exam.



A Summary will be shown and teacher can click finish to finish the create exam wizard.

# Appendix

1. Software Engineering 9th by Somerville, page 73
2. C# Code Style Rules: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions
3. Google JavaScript Style Guide: https://google.github.io/styleguide/jsguide.html
4. Angular - Style Guide: https://angular.io/guide/styleguide#single-responsibility
5. Angular - Architecture overview: https://angular.io/guide/architecture