**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, **19 November 2018** -

Table of Contents

[Table of Contents 2](#_Toc532247135)

[List Of Tables 9](#_Toc532247136)

[List of Figures 13](#_Toc532247137)

[Definitions, Acronyms, and Abbreviations 15](#_Toc532247138)

[A. Introduction 16](#_Toc532247139)

[1. Project Information 16](#_Toc532247140)

[2. Introduction 16](#_Toc532247141)

[3. Current Situation 16](#_Toc532247142)

[4. Problem Definition 17](#_Toc532247143)

[4.1. Proposed Solution 17](#_Toc532247144)

[4.2. Feature Functions 17](#_Toc532247145)

[5. Values and Challenges Value 17](#_Toc532247146)

[6. Functional Requirement 17](#_Toc532247147)

[7. Role and Responsibility 18](#_Toc532247148)

[B. Software Project Management Plan 19](#_Toc532247149)

[1. Problem Definition 19](#_Toc532247150)

[1.1. Name of this Capstone Project 19](#_Toc532247151)

[1.2. Problem Abstract 19](#_Toc532247152)

[1.3. Project Overview 19](#_Toc532247153)

[1.3.1 Current Situation 19](#_Toc532247154)

[1.3.2 Boundaries of the system 19](#_Toc532247155)

[1.3.3 Future Plans 20](#_Toc532247156)

[1.3.4 Development Environment 20](#_Toc532247157)

[1.3.4.1 Hardware requirements 20](#_Toc532247158)

[1.3.4.2 Software requirements 20](#_Toc532247159)

[2. Coding Convention 20](#_Toc532247160)

[3. Project Organization 21](#_Toc532247161)

[3.1. Software Process Model 21](#_Toc532247162)

[3.2. Roles and Responsibilities 22](#_Toc532247163)

[3.3. Tools and Techniques 23](#_Toc532247164)

[4. Project Management Plan 23](#_Toc532247165)

[4.1. Product Backlog 23](#_Toc532247166)

[4.2. Sprint Backlog 24](#_Toc532247167)

[4.2.1 Sprint 1 (10.09.2018 – 23.09.2018): Project initiation 24](#_Toc532247168)

[4.2.1.1 Goal 24](#_Toc532247169)

[4.2.1.2 Development 25](#_Toc532247170)

[4.2.2 Sprint 2 (24.09.2018 – 8.10.2018): Software Document 25](#_Toc532247171)

[4.2.2.1 Goal 25](#_Toc532247172)

[4.2.2.2 Development 25](#_Toc532247173)

[4.2.3 Sprint 3 (9.10.2018 – 23.10.2018): Basic feature and core 26](#_Toc532247174)

[4.2.3.1 Goal 26](#_Toc532247175)

[4.2.3.2 Development 26](#_Toc532247176)

[4.2.4 Sprint 4 (24.10.2018 – 8.11.2018): Software Document 26](#_Toc532247177)

[4.2.4.1 Goal 26](#_Toc532247178)

[4.2.4.2 Development 26](#_Toc532247179)

[4.2.5 Sprint 5 (9.11.2018 – 23.11.2018): Software Document 27](#_Toc532247180)

[4.2.5.1 Goal 27](#_Toc532247181)

[4.2.5.2 Development 27](#_Toc532247182)

[4.2.6 Sprint 6 (24.11.2018 – 13.12.2018): Complete Document 27](#_Toc532247183)

[4.2.6.1 Goal 27](#_Toc532247184)

[4.2.6.2 Development 27](#_Toc532247185)

[4.3. All Meeting minutes 28](#_Toc532247186)

[C. Software Requirement Specification 28](#_Toc532247187)

[1. User Requirement Specification 28](#_Toc532247188)

[1.1. Examination Requirement 28](#_Toc532247189)

[1.2. System Requirement 28](#_Toc532247190)

[2. System Requirement Specification 28](#_Toc532247191)

[2.1. External Interface Requirement 28](#_Toc532247192)

[2.1.1 User Interface 28](#_Toc532247193)

[2.1.2 Hardware Interface 28](#_Toc532247194)

[2.1.3 Software Interface 28](#_Toc532247195)

[2.1.4 Communication Protocol 28](#_Toc532247196)

[2.2. System Overview Diagram 29](#_Toc532247197)

[2.3. List of use case 33](#_Toc532247198)

[2.3.1 <IT> Remove Process Must be killed 33](#_Toc532247199)

[2.3.2 <IT> Add Must Be Killed Process to List 34](#_Toc532247200)

[2.3.3 <IT> Add to List Exception Process 36](#_Toc532247201)

[2.3.4 <IT> Kill Process 39](#_Toc532247202)

[2.3.5 <IT> Get real-time Process 40](#_Toc532247203)

[2.3.6 <IT> Search real-time Process 42](#_Toc532247204)

[2.3.7 <IT> Get Student Screen 43](#_Toc532247205)

[2.3.8 <IT> Submit Student Test Exam 45](#_Toc532247206)

[2.3.9 <IT> Search Online Student 47](#_Toc532247207)

[2.3.10 <IT> Scan student by process 48](#_Toc532247208)

[2.3.11 <Training Department Staff> Add new Semester 50](#_Toc532247209)

[2.3.12 <Leader> Add chapter 52](#_Toc532247210)

[2.3.13 <Leader> Delete chapter 54](#_Toc532247211)

[2.3.14 <Leader> Edit chapter 56](#_Toc532247212)

[2.3.15 <Teacher> Search Question 57](#_Toc532247213)

[2.3.16 <Teacher> Add Question by Form 59](#_Toc532247214)

[2.3.17 <Teacher> Add Question by Import File 62](#_Toc532247215)

[2.3.18 <Teacher> Remove Questions 63](#_Toc532247216)

[2.3.19 <Teacher> Edit Questions 65](#_Toc532247217)

[2.3.20 <Teacher> Approve Test Exam 68](#_Toc532247218)

[2.3.21 <Training Department Staff> Generate Test Exam 70](#_Toc532247219)

[2.3.22 <Training Department Staff> Assign Approver 72](#_Toc532247220)

[3. Software System Attribute 74](#_Toc532247221)

[3.1. Usability 74](#_Toc532247222)

[3.2. Reliability 74](#_Toc532247223)

[3.3. Availability 74](#_Toc532247224)

[3.4. Security 74](#_Toc532247225)

[3.5. Maintainability 74](#_Toc532247226)

[3.6. Portability 74](#_Toc532247227)

[4. Conceptual Diagram 75](#_Toc532247228)

[D. Software Design Description 77](#_Toc532247229)

[1. Design Overview 77](#_Toc532247230)

[2. System Architecture Design 77](#_Toc532247231)

[2.1. Web Server Architecture Design 78](#_Toc532247232)

[2.2. Client Architecture Design 78](#_Toc532247233)

[3. Component Diagram 80](#_Toc532247234)

[4. Detailed Description 81](#_Toc532247235)

[4.1. Class Diagram 81](#_Toc532247236)

[4.2. Class Diagram Explanation 84](#_Toc532247237)

[4.2.1 Semester 84](#_Toc532247238)

[4.2.2 Exam 84](#_Toc532247239)

[4.2.3 ExamStudent 85](#_Toc532247240)

[4.2.4 Student 85](#_Toc532247241)

[4.2.5 BaseTest 86](#_Toc532247242)

[4.2.6 BaseTestContent 87](#_Toc532247243)

[4.2.7 Test 87](#_Toc532247244)

[4.2.8 TestContent 88](#_Toc532247245)

[4.2.9 Question 89](#_Toc532247246)

[4.2.10 Option 89](#_Toc532247247)

[4.2.11 Level 90](#_Toc532247248)

[4.2.12 QuestionType 90](#_Toc532247249)

[4.2.13 QuestionGroup 91](#_Toc532247250)

[4.2.14 QuestionObjective 91](#_Toc532247251)

[4.2.15 Chapter 92](#_Toc532247252)

[4.2.16 Course 92](#_Toc532247253)

[4.2.17 MainObjective 93](#_Toc532247254)

[4.2.18 LearningOutcome 93](#_Toc532247255)

[4.2.19 TeacherTest 94](#_Toc532247256)

[4.2.20 TeacherTestContent 94](#_Toc532247257)

[4.3. Interaction Diagram 95](#_Toc532247258)

[4.3.1 Activity Diagram 95](#_Toc532247259)

[4.3.1.1 Teacher Import Question 95](#_Toc532247260)

[4.3.1.2 Generate Test Exam 97](#_Toc532247261)

[4.3.1.3 Teacher Approve Test Exam 97](#_Toc532247262)

[4.3.1.4 Student Take a Test Exam 98](#_Toc532247263)

[5. Interface 99](#_Toc532247264)

[5.1. Manage Question Bank 100](#_Toc532247265)

[5.2. Import question 101](#_Toc532247266)

[5.3. Matching question 102](#_Toc532247267)

[5.4. Create Test Exam 102](#_Toc532247268)

[5.5. Test Exam List with “Pending” Status 107](#_Toc532247269)

[5.6. Take an Exam 107](#_Toc532247270)

[5.7. Create Exam 110](#_Toc532247271)

[5.8. Set Approver 111](#_Toc532247272)

[6. Database Design 113](#_Toc532247273)

[6.1. Entity Relationship Diagram 113](#_Toc532247274)

[6.2. Data Dictionary 114](#_Toc532247275)

[6.3. ER Diagram Explanation 114](#_Toc532247276)

[6.3.1 Semester 114](#_Toc532247277)

[6.3.2 Exam 115](#_Toc532247278)

[6.3.3 Student 115](#_Toc532247279)

[6.3.4 BaseTest 115](#_Toc532247280)

[6.3.5 BaseTestContent 116](#_Toc532247281)

[6.3.6 Test 116](#_Toc532247282)

[6.3.7 TestContent 117](#_Toc532247283)

[6.3.8 Question 117](#_Toc532247284)

[6.3.9 Option 117](#_Toc532247285)

[6.3.10 Level 117](#_Toc532247286)

[6.3.11 QuestionType 118](#_Toc532247287)

[6.3.12 QuestionGroup 118](#_Toc532247288)

[6.3.13 QuestionObjective 118](#_Toc532247289)

[6.3.14 Chapter 118](#_Toc532247290)

[6.3.15 Course 119](#_Toc532247291)

[6.3.16 MainObjective 119](#_Toc532247292)

[6.3.17 LearningOutcome 119](#_Toc532247293)

[6.3.18 Course Department 119](#_Toc532247294)

[6.3.19 TeacherTest 119](#_Toc532247295)

[6.3.20 TeacherTestContent 120](#_Toc532247296)

[7. Algorithms 121](#_Toc532247297)

[7.1. Calculating Percentage Similarity of 2 Question 121](#_Toc532247298)

[7.1.1 Problem 121](#_Toc532247299)

[7.1.2 Solution 121](#_Toc532247300)

[7.1.3 Complexity 123](#_Toc532247301)

[E. System Implementation & Test 123](#_Toc532247302)

[1. Introduction 123](#_Toc532247303)

[1.1. Overview 123](#_Toc532247304)

[1.2. Test Approach 123](#_Toc532247305)

[2. Database Relationship Diagram 124](#_Toc532247306)

[2.1. Physical Diagram: 124](#_Toc532247307)

[2.2. Data Dictionary: 127](#_Toc532247308)

[2.2.1 Semester 127](#_Toc532247309)

[2.2.2 Exam 127](#_Toc532247310)

[2.2.3 Student 127](#_Toc532247311)

[2.2.4 BaseTest 127](#_Toc532247312)

[2.2.5 BaseTestContent 128](#_Toc532247313)

[2.2.6 Test 128](#_Toc532247314)

[2.2.7 TestContent 129](#_Toc532247315)

[2.2.8 Question 129](#_Toc532247316)

[2.2.9 Option 130](#_Toc532247317)

[2.2.10 Level 130](#_Toc532247318)

[2.2.11 QuestionType 130](#_Toc532247319)

[2.2.12 QuestionGroup 131](#_Toc532247320)

[2.2.13 QuestionObjective 131](#_Toc532247321)

[2.2.14 Chapter 131](#_Toc532247322)

[2.2.15 Course 131](#_Toc532247323)

[2.2.16 MainObjective 132](#_Toc532247324)

[2.2.17 LearningOutcome 132](#_Toc532247325)

[2.2.18 Course Department 132](#_Toc532247326)

[2.2.19 TeacherTest 132](#_Toc532247327)

[2.2.20 TeacherTestContent 133](#_Toc532247328)

[2.2.21 AuthenticationApp 133](#_Toc532247329)

[2.2.22 Import Result 134](#_Toc532247330)

[2.2.23 DefaultProcess 134](#_Toc532247331)

[3. Performance Measures 134](#_Toc532247332)

[3.1. Parsing Question 134](#_Toc532247333)

[3.2. Validate Parsed Questions 134](#_Toc532247334)

[3.3. Matching Questions of a Course 134](#_Toc532247335)

[3.4. Random Question to Create Exam Base on Chapters/ Learning Outcome 134](#_Toc532247336)

[3.5. Get Exam Test from Server 135](#_Toc532247337)

[3.6. Submit Exam Tests of a Class 135](#_Toc532247338)

[4. Test Plan 135](#_Toc532247339)

[4.1. Features to be tested 135](#_Toc532247340)

[4.2. Features not to be tested 135](#_Toc532247341)

[5. System Testing Test Case 135](#_Toc532247342)

[5.1. Manage Exam System Test Case 135](#_Toc532247343)

[5.1.1 Import Question 135](#_Toc532247344)

[6. System Implement 141](#_Toc532247345)

[6.1. Web Server Application 141](#_Toc532247346)

[6.2. Web Client Application 142](#_Toc532247347)

[F. Software User’s Manual 143](#_Toc532247348)

[1. Installing Guide 143](#_Toc532247349)

[2. User Guide 146](#_Toc532247350)

[2.1. Admin Create Semester 146](#_Toc532247351)

[2.2. Teacher approve/reject exam 149](#_Toc532247352)

[2.3. Staff Create Exam 151](#_Toc532247353)

[G. Appendix 156](#_Toc532247354)

List Of Tables

[Table 1: Definitions, Acronyms, and Abbreviations 15](#_Toc532247355)

[Table 2: Role and Responsibilities 18](#_Toc532247356)

[Table 3: Hardware Requirements 20](#_Toc532247357)

[Table 4 Software Requirement 20](#_Toc532247358)

[Table 5 Roles and Responsibilities Detail 22](#_Toc532247359)

[Table 6 Tools and Techniques 23](#_Toc532247360)

[Table 7 Product Backlog 24](#_Toc532247361)

[Table 8 Sprint 1 Development 25](#_Toc532247362)

[Table 9 SPRINT 2 DEVELOPMENT 26](#_Toc532247363)

[Table 10 SPRINT 3 DEVELOPMENT 26](#_Toc532247364)

[Table 11 SPRINT 4 DEVELOPMENT 27](#_Toc532247365)

[Table 12 SPRINT 5 DEVELOPMENT 27](#_Toc532247366)

[Table 13 SPRINT 6 DEVELOPMENT 27](#_Toc532247367)

[Table 14 Remove process must be killed 34](#_Toc532247368)

[Table 15 Add Must Be Killed Process to List 36](#_Toc532247369)

[Table 16 ADD TO LIST EXCEPTION PROCESS 38](#_Toc532247370)

[Table 17 Kill Process 40](#_Toc532247371)

[Table 18 <IT> GET REALTIME PROCESS 41](#_Toc532247372)

[Table 19 <IT> SEARCH REALTIME PROCESS 43](#_Toc532247373)

[Table 20 <IT> GET STUDENT SCREEN 45](#_Toc532247374)

[Table 21 <IT> Submit Student Test Exam 47](#_Toc532247375)

[Table 22 <IT> SEARCH ONLINE STUDENT 48](#_Toc532247376)

[Table 23 <IT> Scan student by process 50](#_Toc532247377)

[Table 24 <TRAINING DEPARTMENT STAFF> ADD NEW SEMESTER 52](#_Toc532247378)

[Table 25 Add chapter 54](#_Toc532247379)

[Table 26 Delete chapter 55](#_Toc532247380)

[Table 27 <Leader> Edit chapter 57](#_Toc532247381)

[Table 29 <Teacher> Search Question 59](#_Toc532247382)

[Table 30 <TEACHER> ADD QUESTION BY FORM 61](#_Toc532247383)

[Table 31 <Teacher> Add Question by Import File 63](#_Toc532247384)

[Table 32 <Teacher> Remove Questions 65](#_Toc532247385)

[Table 33 <Teacher> Edit Questions 67](#_Toc532247386)

[Table 34 <Teacher> Approve Test Exam 70](#_Toc532247387)

[Table 35 <Training department staff> Generate Test Exam 72](#_Toc532247388)

[Table 36 <Training department staff> Assign Approver 74](#_Toc532247389)

[Table 37 CONCEPTUAL DIAGRAM EXPLAINATION 76](#_Toc532247390)

[Table 38 ARCHITECTURE OVERVIEW DESCRIPTION 78](#_Toc532247391)

[Table 39 Component diagram dictionary 81](#_Toc532247392)

[Table 40 Class diagram Description 84](#_Toc532247393)

[Table 41 Class Semester Explanation 84](#_Toc532247394)

[Table 42 Class Semester method explanation 84](#_Toc532247395)

[Table 43 Class exam Explanation 85](#_Toc532247396)

[Table 44 Class exam method explanation 85](#_Toc532247397)

[Table 45 Class ExamStudent Explanation 85](#_Toc532247398)

[Table 46 Class ExamStudent method explanation 85](#_Toc532247399)

[Table 47 CLASS STUDENT EXPLANATION 86](#_Toc532247400)

[Table 48 Class student method explanation 86](#_Toc532247401)

[Table 49 CLASS BASETEST EXPLANATION 86](#_Toc532247402)

[Table 50 Class basetest method explanation 87](#_Toc532247403)

[Table 51 CLASS BASETESTCONTENT EXPLANATION 87](#_Toc532247404)

[Table 52 Class basetestcontent method explanation 87](#_Toc532247405)

[Table 53 CLASS TEST EXPLANATION 88](#_Toc532247406)

[Table 54 Class test method explanation 88](#_Toc532247407)

[Table 55 CLASS TESTCONTENT EXPLANATION 88](#_Toc532247408)

[Table 56 Class testcontent method explanation 89](#_Toc532247409)

[Table 57 CLASS QUESTION EXPLANATION 89](#_Toc532247410)

[Table 58 Class question method explanation 89](#_Toc532247411)

[Table 59 CLASS OPTION EXPLANATION 90](#_Toc532247412)

[Table 60 Class option method explanation 90](#_Toc532247413)

[Table 61 CLASS level EXPLANATION 90](#_Toc532247414)

[Table 62 Class level method explanation 90](#_Toc532247415)

[Table 63 CLASS QUESTIONTYPE EXPLANATION 91](#_Toc532247416)

[Table 64 Class questiontype method explanation 91](#_Toc532247417)

[Table 65 CLASS QUESTIONGROUP EXPLANATION 91](#_Toc532247418)

[Table 66 Class questiongroup method explanation 91](#_Toc532247419)

[Table 67 CLASS QUESTIONOBJECTIVE EXPLANATION 91](#_Toc532247420)

[Table 68 Class questionobjective method explanation 92](#_Toc532247421)

[Table 69 CLASS CHAPTER EXPLANATION 92](#_Toc532247422)

[Table 70 Class chapter method explanation 92](#_Toc532247423)

[Table 71 CLASS COURSE EXPLANATION 92](#_Toc532247424)

[Table 72 Class course method explanation 93](#_Toc532247425)

[Table 73 CLASS MAINOBJECTIVE EXPLANATION 93](#_Toc532247426)

[Table 74 Class mainobjective method explanation 93](#_Toc532247427)

[Table 75 CLASS LEARNINGOUTCOME EXPLANATION 93](#_Toc532247428)

[Table 76 Class learningoutcome method explanation 94](#_Toc532247429)

[Table 77 CLASS TeacherTest EXPLANATION 94](#_Toc532247430)

[Table 78 Class teachertest method explanation 94](#_Toc532247431)

[Table 79 CLASS TeacherTestContent EXPLANATION 95](#_Toc532247432)

[Table 80 Class teachertestcontent method explanation 95](#_Toc532247433)

[Table 81 manage question bank fields 100](#_Toc532247434)

[Table 82 manage question bank buttons 100](#_Toc532247435)

[Table 83 import question buttons 101](#_Toc532247436)

[Table 84 matching question 102](#_Toc532247437)

[Table 85create test exam fields 106](#_Toc532247438)

[Table 86 create test exam buttons 106](#_Toc532247439)

[Table 87 test exam list with “pending” status buttons 107](#_Toc532247440)

[Table 88 take an exam fields 109](#_Toc532247441)

[Table 89 Take an exam buttons 110](#_Toc532247442)

[Table 90 create exam fields 111](#_Toc532247443)

[Table 91 create exam buttons 111](#_Toc532247444)

[Table 92 set approver fields 112](#_Toc532247445)

[Table 93 set approver buttons 112](#_Toc532247446)

[Table 94 ERD Dictionary 114](#_Toc532247447)

[Table 95 Semester table Description 115](#_Toc532247448)

[Table 96 Exam table Description 115](#_Toc532247449)

[Table 97 Student table Description 115](#_Toc532247450)

[Table 98 BaseTest table Description 116](#_Toc532247451)

[Table 99 BaseTestContent table Description 116](#_Toc532247452)

[Table 100 Test table Description 116](#_Toc532247453)

[Table 101 TestContent table Description 117](#_Toc532247454)

[Table 102 Question table Description 117](#_Toc532247455)

[Table 103 Option table Description 117](#_Toc532247456)

[Table 104 Level table Description 118](#_Toc532247457)

[Table 105 QuestionType table Description 118](#_Toc532247458)

[Table 106 QuestionGroup table Description 118](#_Toc532247459)

[Table 107 QuestionObjective table Description 118](#_Toc532247460)

[Table 108 Chapter table Description 118](#_Toc532247461)

[Table 109 Course table Description 119](#_Toc532247462)

[Table 110 MainObjective table Description 119](#_Toc532247463)

[Table 111 LearningOutcome table Description 119](#_Toc532247464)

[Table 112 CourseDepartment table Description 119](#_Toc532247465)

[Table 113 TeacherTest table Description 120](#_Toc532247466)

[Table 114 TeacherTestContent table Description 120](#_Toc532247467)

[Table 115 Physical Diagram Description 126](#_Toc532247468)

[Table 116 SEMESTER TABLE DESCRIPTION 127](#_Toc532247469)

[Table 117 Exam table Description 127](#_Toc532247470)

[Table 118 Student table Description 127](#_Toc532247471)

[Table 119 BaseTest table Description 128](#_Toc532247472)

[Table 120 BaseTestContent table Description 128](#_Toc532247473)

[Table 121 Test table Description 129](#_Toc532247474)

[Table 122 TestContent table Description 129](#_Toc532247475)

[Table 123 Question table Description 130](#_Toc532247476)

[Table 124 Option table Description 130](#_Toc532247477)

[Table 125 Level table Description 130](#_Toc532247478)

[Table 126 QuestionType table Description 131](#_Toc532247479)

[Table 127 QuestionGroup table Description 131](#_Toc532247480)

[Table 128 QuestionObjective table Description 131](#_Toc532247481)

[Table 129 Chapter table Description 131](#_Toc532247482)

[Table 130 Course table Description 131](#_Toc532247483)

[Table 131 MainObjective table Description 132](#_Toc532247484)

[Table 132 LearningOutcome table Description 132](#_Toc532247485)

[Table 133 CourseDepartment table Description 132](#_Toc532247486)

[Table 134 TeacherTest table Description 133](#_Toc532247487)

[Table 135 TeacherTestContent table Description 133](#_Toc532247488)

[Table 136 AuthenticationApp table Description 133](#_Toc532247489)

[Table 137 ImportResult table Description 134](#_Toc532247490)

[Table 138 DefaultProcess table Description 134](#_Toc532247491)

[Table 139 Import Question Test Result 1 136](#_Toc532247492)

[Table 140 Import question test result 2 137](#_Toc532247493)

[Table 141 Import question test result 3 137](#_Toc532247494)

[Table 142 Matching Question test result 137](#_Toc532247495)

[Table 143 Generate Test Result 138](#_Toc532247496)

[Table 144 Approve Exam Test result 138](#_Toc532247497)

[Table 145 Run exam tool test result 1 139](#_Toc532247498)

[Table 146 Run exam tool test result 2 139](#_Toc532247499)

[Table 147 Run exam tool test result 3 139](#_Toc532247500)

[Table 148 Take Exam test result 1 140](#_Toc532247501)

[Table 149 Take Exam test result 2 140](#_Toc532247502)

[Table 150 Take Exam test result 3 141](#_Toc532247503)

List of Figures

[Figure 1 The Scrum Process 22](#_Toc532247504)

[Figure 2: Use case overview Examination System 29](#_Toc532247505)

[Figure 3: Teacher use case Detail 30](#_Toc532247506)

[Figure 4: Leader use case detail 30](#_Toc532247507)

[Figure 5: IT use case detail 31](#_Toc532247508)

[Figure 6: Manager use case detail 31](#_Toc532247509)

[Figure 7: Student use case detail 31](#_Toc532247510)

[Figure 8: Training Department Staff use case detail 32](#_Toc532247511)

[Figure 10 Add Must Be Killed Process to List 34](#_Toc532247512)

[Figure 11 Add to List Exception Process 36](#_Toc532247513)

[Figure 12 Kill Process 39](#_Toc532247514)

[Figure 13 <IT> Get real-time Process 40](#_Toc532247515)

[Figure 14 <IT> Search real-time Process 42](#_Toc532247516)

[Figure 15 <IT> Get Student Screen 43](#_Toc532247517)

[Figure 16 <IT> Submit Student Test Exam 45](#_Toc532247518)

[Figure 17 <IT> Search Online Student 47](#_Toc532247519)

[Figure 13 <IT> Scan student by process 49](#_Toc532247520)

[Figure 18 <Training Department Staff> Add new Semester 50](#_Toc532247521)

[Figure 22 Add chapter 52](#_Toc532247522)

[Figure 23 Delete chapter 54](#_Toc532247523)

[Figure 24 <Leader> Edit chapter 56](#_Toc532247524)

[Figure 25 <Teacher> Search Question 58](#_Toc532247525)

[Figure 26 <Teacher> Add Question by Form 60](#_Toc532247526)

[Figure 27 <Teacher> Add Question by Import File 62](#_Toc532247527)

[Figure 28 <Teacher> Remove Questions 64](#_Toc532247528)

[Figure 29 <Teacher> Edit Questions 65](#_Toc532247529)

[Figure 31 <Teacher> Approve Test Exam 68](#_Toc532247530)

[Figure 32 <Teacher> generate Test Exam 70](#_Toc532247531)

[Figure 33 < Training Department Staff > Approve Test Exam 72](#_Toc532247532)

[Figure 34 Conceptual Diagram 75](#_Toc532247533)

[Figure 35 System Architecture Design 77](#_Toc532247534)

[Figure 36 Architecture Overview 78](#_Toc532247535)

[Figure 37 Web Admin MVC Architecture 79](#_Toc532247536)

[Figure 38 Component Diagram 80](file:///D:\Source\CapstoneProject_1\document\ExamTool.docx#_Toc532247537)

[Figure 39 Class diagram 82](file:///D:\Source\CapstoneProject_1\document\ExamTool.docx#_Toc532247538)

[Figure 40 Activity diagram Teacher Import Question 97](#_Toc532247539)

[Figure 41 Activity diagram Training department staff generate Test Exam 97](#_Toc532247540)

[Figure 42 Activity diagram Approve Test Exam 98](#_Toc532247541)

[Figure 43 Activity diagram Student take a test exam 99](#_Toc532247542)

[Figure 44 manage question bank 100](#_Toc532247543)

[Figure 45 import question 101](#_Toc532247544)

[Figure 46 matching question 102](#_Toc532247545)

[Figure 47 create test exam 103](#_Toc532247546)

[Figure 48 create test exam 103](#_Toc532247547)

[Figure 49 create test exam 104](#_Toc532247548)

[Figure 50 create test exam 104](#_Toc532247549)

[Figure 51 create test exam 105](#_Toc532247550)

[Figure 52 create test exam 105](#_Toc532247551)

[Figure 53 test exam list with "pending" status 107](#_Toc532247552)

[Figure 54 take an exam 108](#_Toc532247553)

[Figure 55 take an exam 108](#_Toc532247554)

[Figure 56 take an exam 109](#_Toc532247555)

[Figure 57 take an exam 109](#_Toc532247556)

[Figure 58 create exam 111](#_Toc532247557)

[Figure 59 set approver 111](#_Toc532247558)

[Figure 60 ER DIAGRAM 113](#_Toc532247559)

[Figure 61 Levenshtein distance example result 123](#_Toc532247560)

[Figure 62 Physical diagram 125](file:///D:\Source\CapstoneProject_1\document\ExamTool.docx#_Toc532247561)

[Figure 63 Onion Architecture 142](#_Toc532247562)

[Figure 64 Event-driven architecture 142](#_Toc532247563)

[Figure 65 Angular Component Example 143](#_Toc532247564)

[Figure 66 Guide create semester 1 146](#_Toc532247565)

[Figure 67 Guide create semester 2 146](#_Toc532247566)

[Figure 68 Guide create semester 3 147](#_Toc532247567)

[Figure 69 Guide create semester 4 147](#_Toc532247568)

[Figure 70 Guide create semester 5 148](#_Toc532247569)

[Figure 71 Guide teacher approve test exam 149](#_Toc532247570)

[Figure 72 Guide teacher approve test exam 2 149](#_Toc532247571)

[Figure 73 Guide teacher approve test exam 3 150](#_Toc532247572)

[Figure 74 Guide teacher approve test exam 4 150](#_Toc532247573)

[Figure 75 Guide teacher approve test exam 5 151](#_Toc532247574)

[Figure 76 Guide teacher approve test exam 6 151](#_Toc532247575)

[Figure 77 Guide Staff create exam 1 152](#_Toc532247576)

[Figure 78 GUIDE STAFF CREATE EXAM 2 152](#_Toc532247577)

[Figure 79 GUIDE STAFF CREATE EXAM 3 153](#_Toc532247578)

[Figure 80 GUIDE STAFF CREATE EXAM 4 153](#_Toc532247579)

[Figure 81 GUIDE STAFF CREATE EXAM 5 154](#_Toc532247580)

[Figure 82 GUIDE STAFF CREATE EXAM 6 154](#_Toc532247581)

[Figure 83 GUIDE STAFF CREATE EXAM 7 155](#_Toc532247582)

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. L.O 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 L.O will ensure the coverage of a course content inside it. Beside the insurance of the test exam content, we also ensure the test exam will be protected by Block-chain technology. Block-chain is a technology which allows data to be transmitted safely and correctly by a very complex encryption. It 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 exam 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 test exam. 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 update the question 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 exam.

## Problem Definition

### Proposed Solution

* The course is divided by topic and chapter, not cover all the content of a course.
* The question updating is repeated and not synchronized. So that, reduce test exam quality by duplicate question and waste of server’s storage.
* Cannot manage student take the exam process during examinations

### Feature Functions

* **Random question base on L.O:** To generate test exam, staff will select L.O first, then all questions will be random base on them.
* **Update question in question bank:** After lecture edit questions in the test exam, that questions in question bank will be updated.
* **Anti-cheating:** We will prevent unnecessary actions and all processes in the list which was not necessary during the test, 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 test exam process has problem by connection, etc.

**IT**

* IT can manage student process

**System Handler**

* Reduced cheating
* Explicit generate test exam, exam, approve process by applying block chain.
* Using authenticate service from FPT University

## 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 to prevent cheating and block-chain technology 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 block-chain in approved test exam process.

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 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 L.O or chapters 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.
* Kill process real-time.
* ExamTool run best for Window 10.

*Our system hasn’t supported:*

* ExamTool is not available 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.
* Support all question type (reading, listen, writing…)

#### 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.

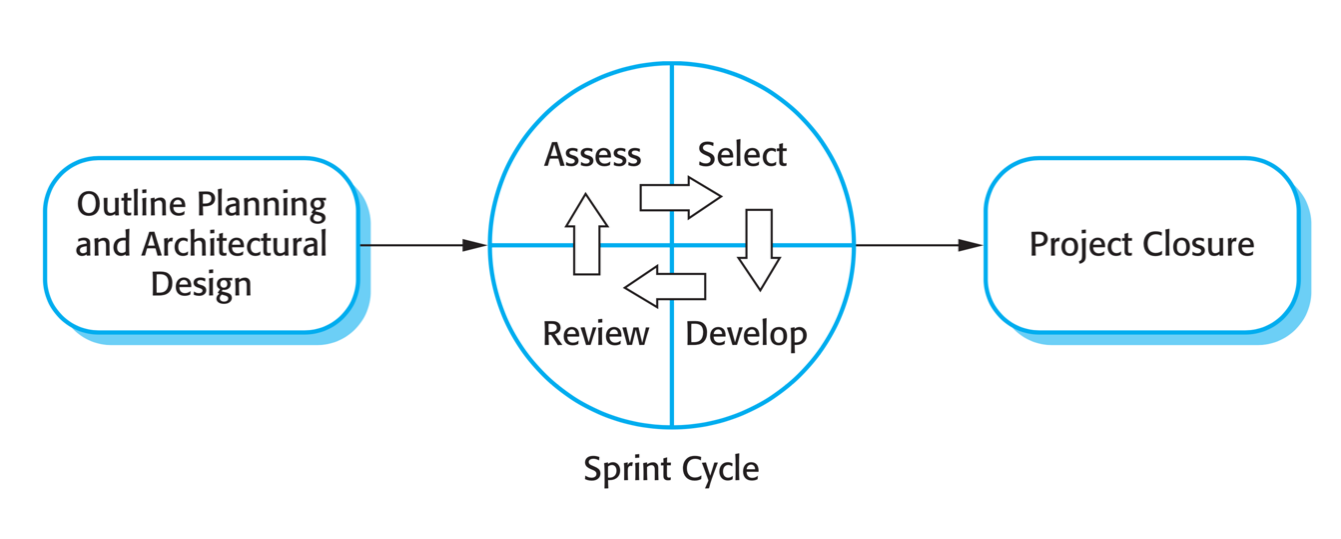


Figure 1 The Scrum Process

*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 |
| 2 | 4 | Software Requirement Specification | 4.1 | User Requirement Specification |
|  |  |  | 4.2 | Software System attribute |
|  |  |  | 4.3 | Conceptual diagram |
|  | 5 | Software Design Diagram | 5.1 | Design Overview |
|  |  |  | 5.2 | System Architectural Design |
|  |  |  | 5.3 | Component Diagram |
|  |  |  | 5.4 | Interface |
|  |  |  | 5.5 | Interface Design |
|  |  |  | 5.6 | Algorithms |
| 3 | 6 | Anti duplicate question algorithms | 6.1 | System Problem definition |
|  |  |  | 6.2 | Implement Algorithms |
|  | 7 | Teacher import question exported from moodle system | 7.1 | Import question from Moodle |
|  |  |  | 7.2 | Manage Question |
|  |  |  | 7.3 | Manage Course |
|  |  |  | 7.4 | Manage Exam |
| 4 | 8 | Leader Manage LO | 8.1 | Manage LO |
|  |  |  | 8.2 | Manage Chapter |
|  | 9 | Manage Test Exam | 9.1 | Staff Generate Test Exam |
|  |  |  | 9.2 | Leader Approve Test Exam |
|  |  |  | 9.3 | Teacher Review Student Test Exam |
|  |  |  | 9.4 | Blockchain Implement |
|  |  |  | 9.5 | Manage public test exam |
|  | 10 | Manage Student take Test Exam Process | 10.1 | Student App Implement |
|  |  |  | 10.2 | Real-time kill process |
|  |  |  | 10.3 | Student take an exam |
| 5 | 11 | Get Student Screen | 11.1 | Get Student Screen |
|  | 12 | FPT Authenticate merge | 12.1 | Use FPT authenticate |
|  | 13 | Test System | 13.1 | Run test with FPT student |
|  |  |  | 13.2 | Report bug |
|  |  |  | 13.3 | Fix bug |
|  |  |  | 13.4 | Test document |
|  | 14 | Software User’s Manual | 14.1 | Installing guide |
|  |  |  | 14.2 | User manual |
|  | 15 | Document paper | 1.5 | Paper document |

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

4 User Requirement Specification

4.1 Software Requirement Specification

4.2 Software System attribute

4.3 Conceptual diagram

5.1 Design Overview

5.2 System Architectural Design

5.3 Component Diagram

5.4 Detailed Description

5.5 Interface

5.6 Interface Design

5.7 Algorithms

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 4 | User Requirement Specification | HungNM |
| 4.1 | Software Requirement Specification | HungNM |
| 4.2 | Software System attribute | HungNM |
| 4.3 | Conceptual diagram | HieuCT |
| 5.1 | Design Overview | HieuCT |
| 5.2 | System Architectural Design | SangTT |
| 5.3 | Component Diagram | SangTT |
| 5.4 | Detailed Description | HieuCT |
| 5.5 | Interface | HungNM |
| 5.6 | Interface Design | HieuCT, HungNM |
| 5.7 | Algorithms | HieuCT, SangTT |

Table 9 SPRINT 2 DEVELOPMENT

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

##### Goal

6.1 System Problem definition

6.2 Implement Algorithms

7.1 Import question from Moodle

7.2 Manage Exam

7.3 Manage Course

7.4 Manage Question

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 6.1 | System Problem definition | HieuCT |
| 6.2 | Implement Algorithms | HieuCT |
| 7.1 | Import question from Moodle | HungNM |
| 7.2 | Manage Exam | HungNM, SangTT |
| 7.3 | Manage Course | HungNM, SangTT |
| 7.4 | Manage Question | HungNM, SangTT |

Table 10 SPRINT 3 DEVELOPMENT

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

##### Goal

7.4 Teacher Import and Manage Question

8.1 Manage LO

8.2 Manage Chapter

9.1 Staff generate test exam

10.1 Student App Implement

10.2 Real-time kill Process Implement

9.2 Leader Approve Test Exam

10.3 Student Take Exam

9.3 Review Student Test Exam

9.4 Block-chain Implement

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 7.4 | Teacher Import and Manage Question | HungNM |
| 8.1 | Manage LO | SangTT, HungNM |
| 8.2 | Manage Chapter | SangTT, HungNM |
| 9.1 | Staff generate test exam | SangTT, HungNM |
| 10.1 | Student App Implement | HieuCT |
| 10.2 | Real-time kill Process Implement | HieuCT |
| 9.2 | Leader Approve Test Exam | SangTT, HungNM |
| 10.3 | Student Take Exam | HieuCT |
| 9.3 | Review Student Test Exam | HieuCT |
| 9.4 | Block-chain Implement | SangTT |

Table 11 SPRINT 4 DEVELOPMENT

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

##### Goal

11.1 IT Get Student Screen

12.1 Login with FPT API

9.5 Manager public test Exam

13.1 Run test With FPT Student

13.2 Report bug

13.3 Fix bug

13.4 Test document

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 11.1 | IT Get Student Screen | HieuCT |
| 12.1 | Login with FPT API | SangTT, HieuCT |
| 9.5 | Manager public test Exam | SangTT, HungNM |
| 13.1 | Run test With FPT Student | SangTT, HieuCT |
| 13.2 | Report bug | HieuCT |
| 13.3 | Fix bug | HieuCT |
| 13.4 | Test document | SangTT, HungNM |

Table 12 SPRINT 5 DEVELOPMENT

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

##### Goal

14.1 Installation Guide

14.2 User Manual

15.1 Paper Document

##### Development

|  |  |  |
| --- | --- | --- |
| **Task ID** | **Task** | **Responsible** |
| 14.1 | Installation Guide | HieuCT |
| 14.2 | User Manual | SangTT, HungNM |
| 15.1 | Paper Document | SangTT, HungNM |

Table 13 SPRINT 6 DEVELOPMENT

### All Meeting minutes

All meeting minutes saved at: “https://trello.com/capstonproject2”

# 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 Wi-Fi automatically when disconnected during exam.

### System Requirement

* + - * 1. The system does the following functions:
* Categorize questions by L.O.
* Secure and ensure exam creation process step by step by using block-chain 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.

## 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

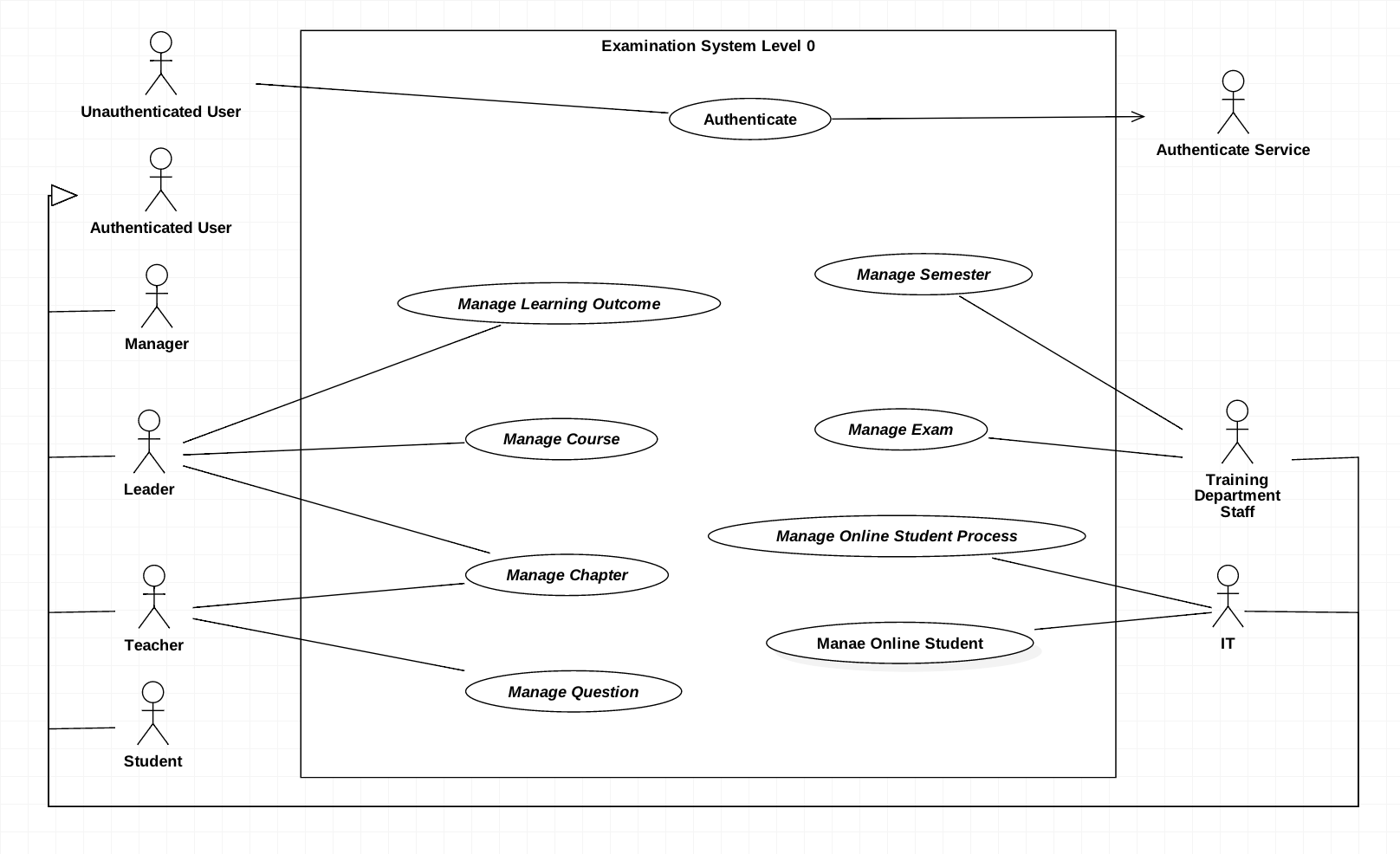
****

Figure 2: Use case overview Examination System

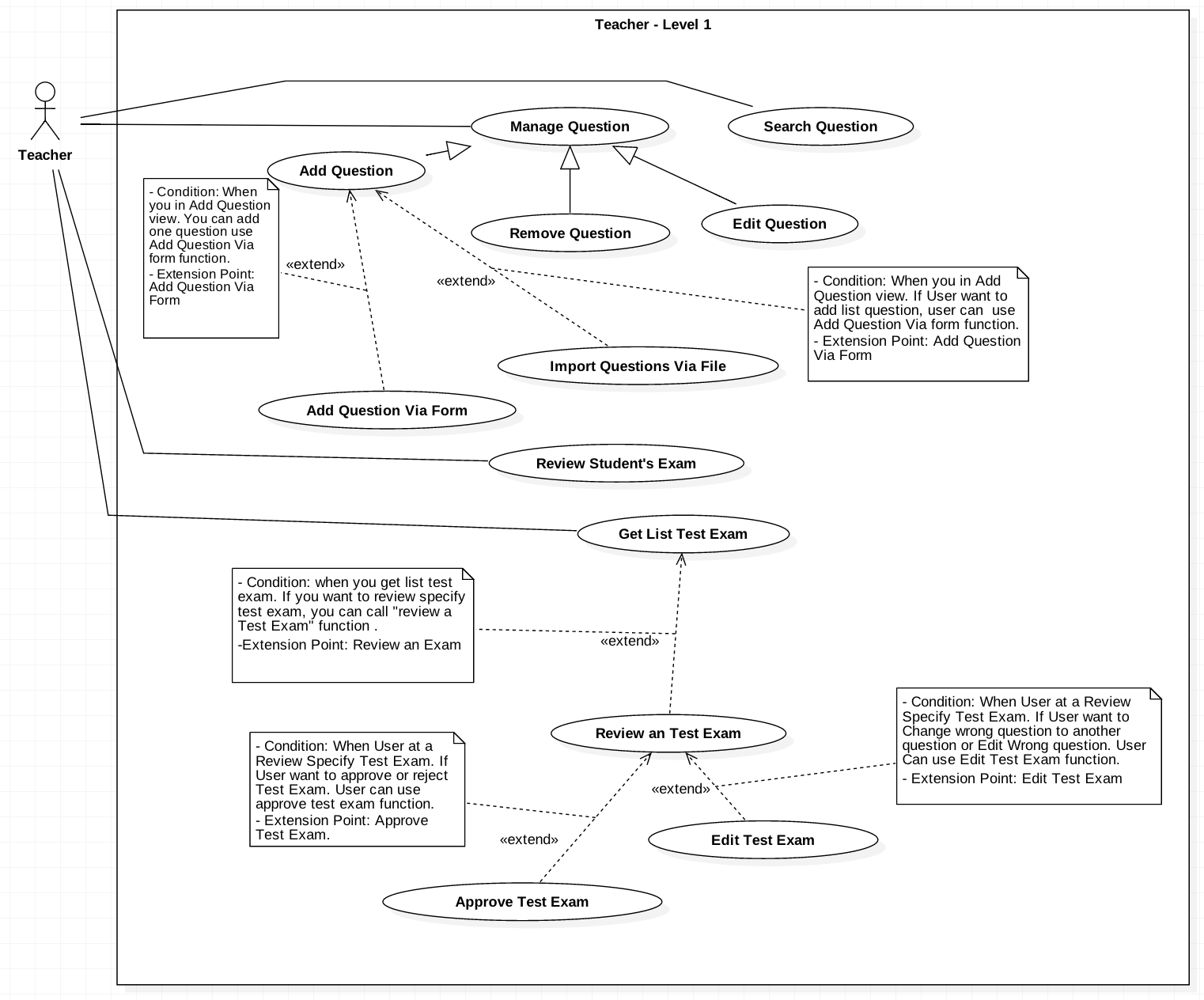


Figure 3: Teacher use case Detail

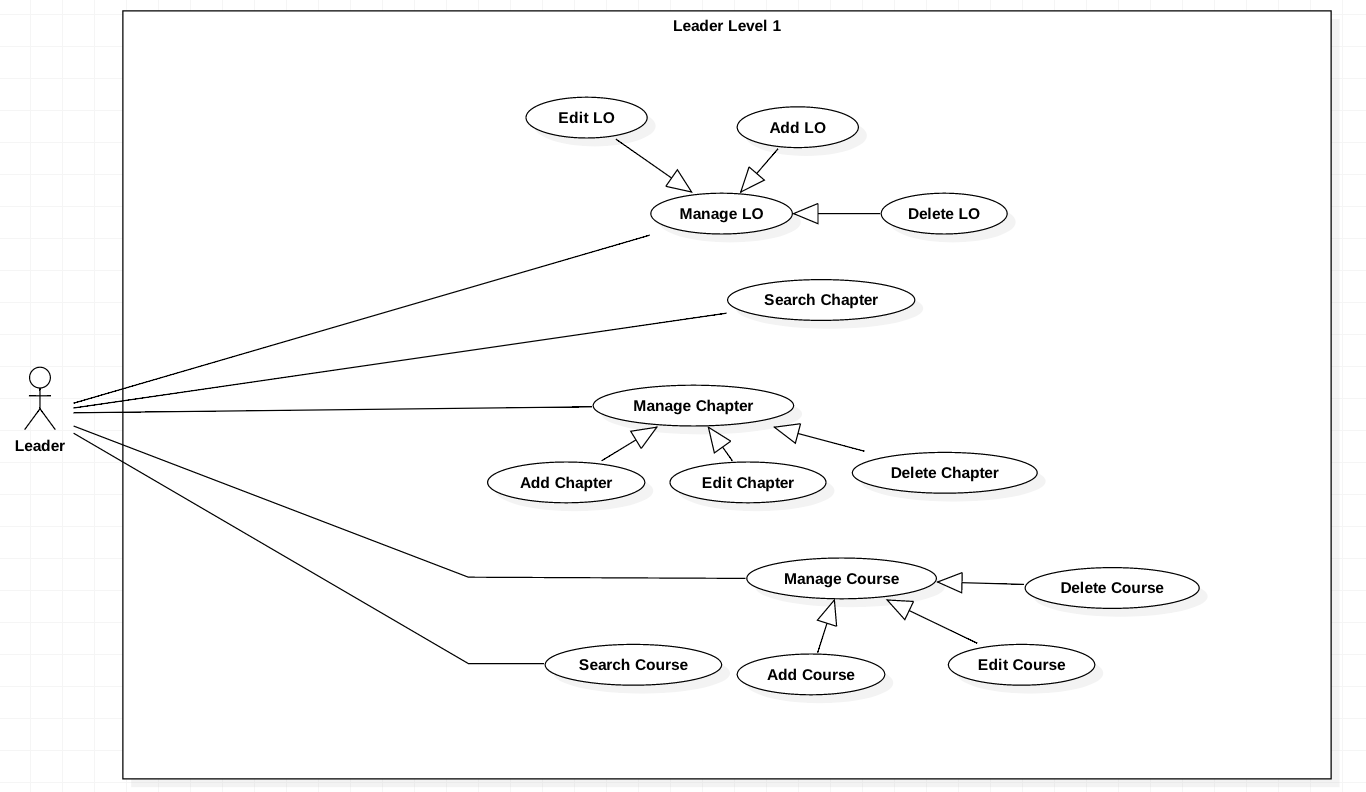
******

Figure 4: Leader use case detail

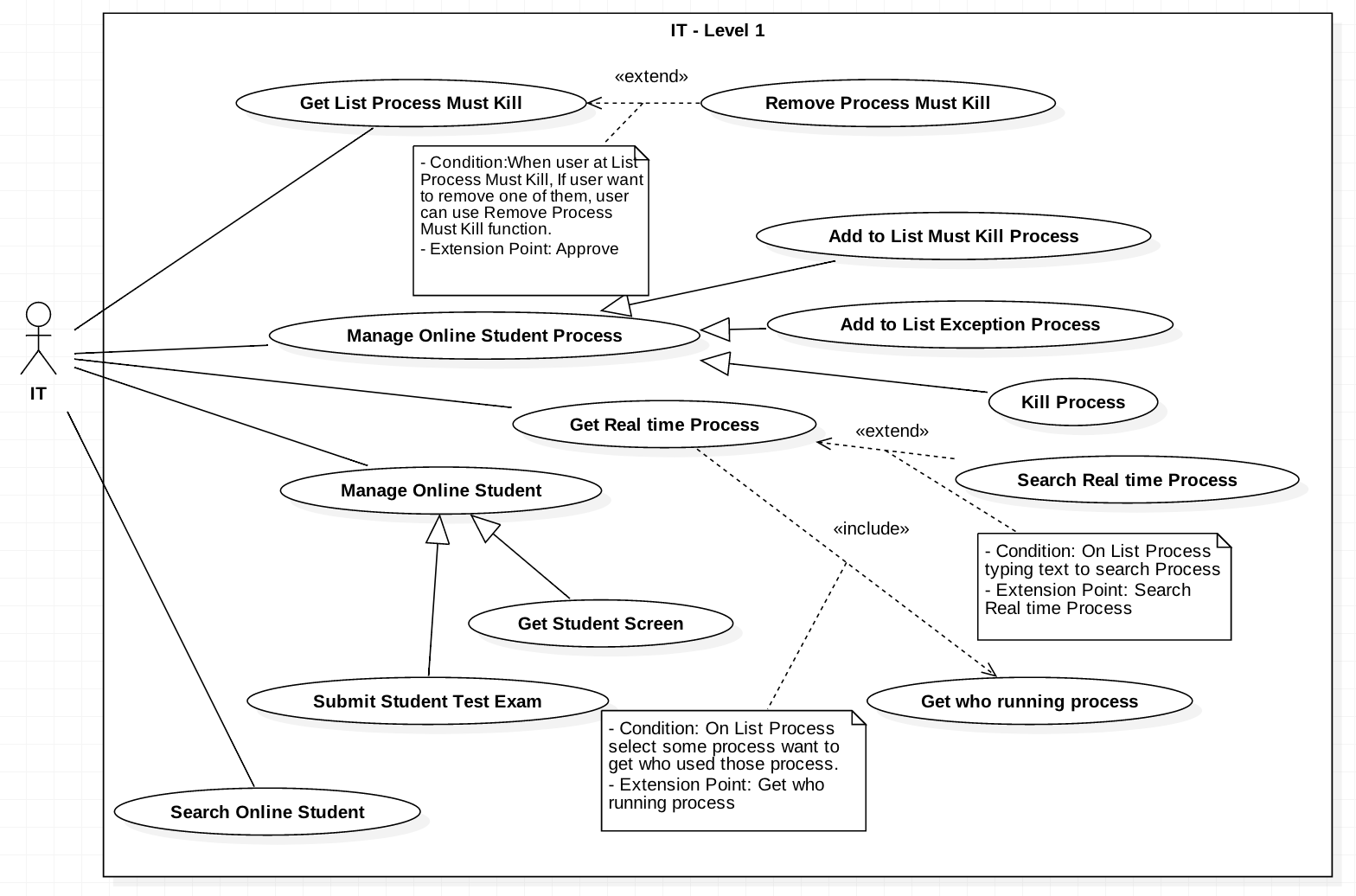
******

Figure 5: IT use case detail

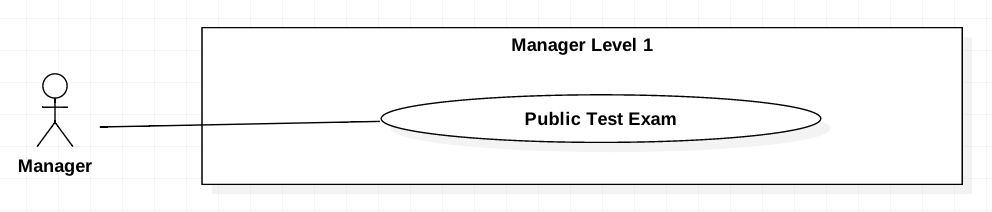


Figure 6: Manager use case detail

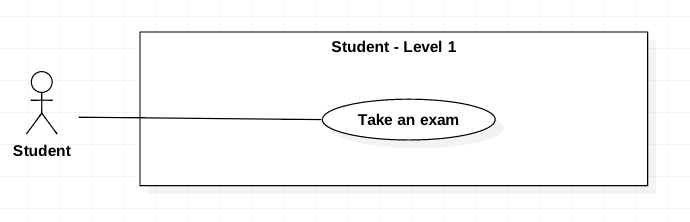
******

Figure 7: Student use case detail

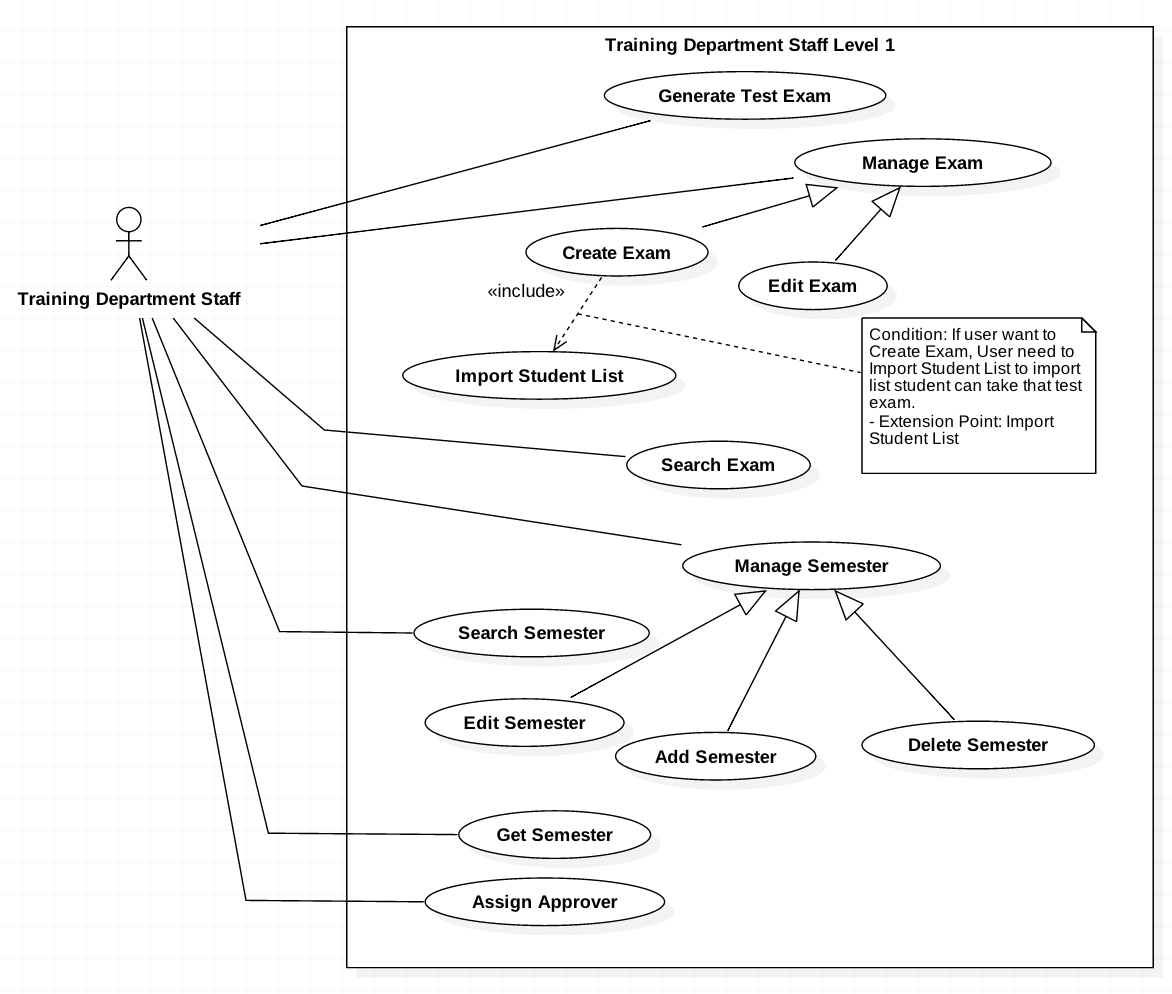
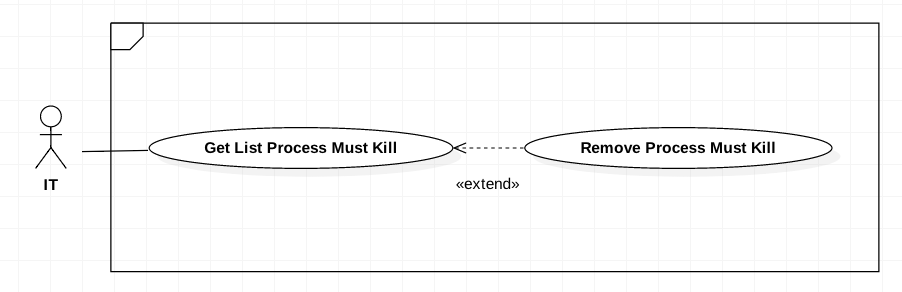


Figure 8: Training Department Staff use case detail

### List of use case

#### <IT> 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:*   * *Process name* * *Application name* * *Process path* | | *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:**   * *Get all process must kill available in system (A).* * *filter process id selected from A by user (B).* * *Remove process in A which have id contain in B.* | | | |

Table 14 Remove process must be killed

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

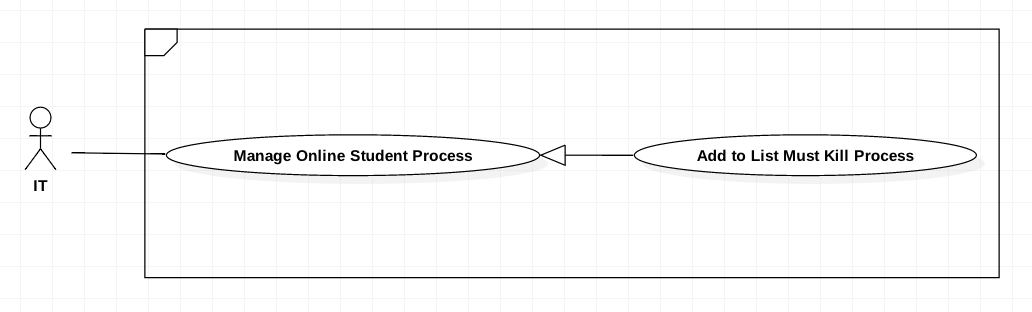


Figure 10 Add Must Be Killed Process to List

|  |  |  |  |
| --- | --- | --- | --- |
| **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:*   * *Process name* * *Application name* * *Process path* | | *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 processes must killed. System check if process information is missing*  *[Exception]* | | *5* |  | *The system will send Kill Process command to student’s application which have Process Name equal added Process Name* |   **Exceptions:**   |  |  |  | | --- | --- | --- | | **Step** | **Cause** | **System Response** | | *1* | *Process information is missing* | *Show message to IT client* |   **Relationships: N/A**  **Business Rules:**   * *When student start the Exam Client up, IT Managers can get the real time processes of that student (A).* * *Base on A, they can add any of those processes to the must kill list by select process from A (B) and call add to process must kill command.* * Push B’s information to list process must kill. * B’s information contains:   + Process Name   + Application Name (Application run that process)   + Process Path (Path of application running that process) * After add processes to list process must kill. Kill all process in A which contain Process Name in B. | | | |

Table 15 Add Must Be Killed Process to List

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

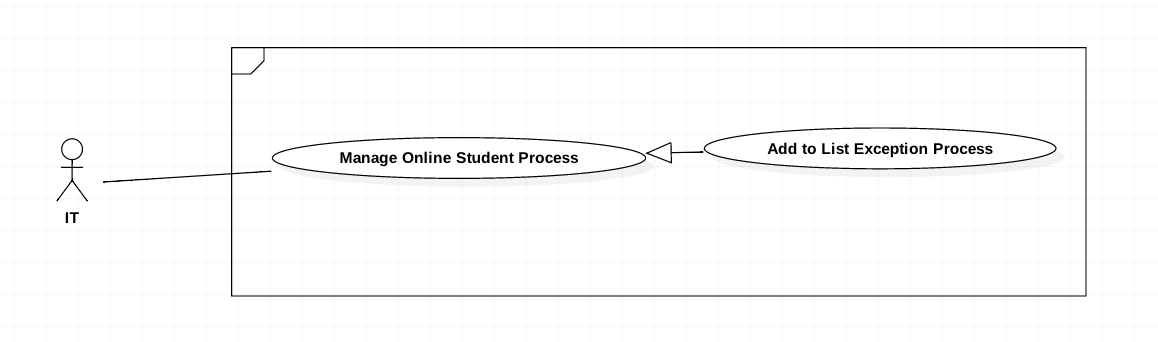


Figure 11 Add to List Exception Process

|  |  |  |  |
| --- | --- | --- | --- |
| **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 computer‘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:*   * *Process Name* * *Application Name* * *Process Path* | | *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: N/A**  **Business Rules:**   * *Get the user input’s process information (A). A contain:*    + *Process Name*   + *Application Name (Application run that process)*   + *Process Path (Path of application running that process)* * *Get all process storage in system which type is exception process (B).* * *Check A if not existed in B, then insert A to system.* | | | |

Table 16 ADD TO LIST EXCEPTION PROCESS

#### <IT> Kill Process

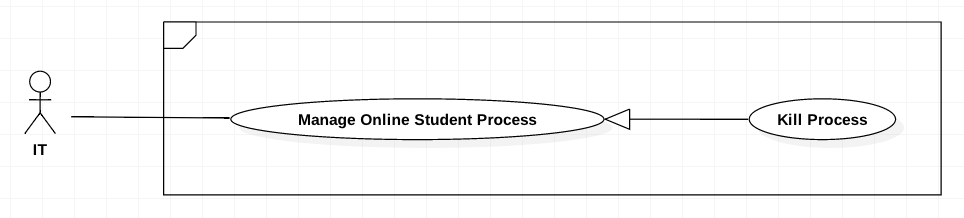


Figure 12 Kill Process

|  |  |  |  |
| --- | --- | --- | --- |
| **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:*   * *Process Name* * *Application Name* * *Process Path* | | *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: N/A**  **Business Rules:**   * *When student start the Exam Client up, IT Managers get the real time processes of that student (A).* * *Base on A, Get all process’s name of selected process by user (B).* * *Kill process in A which have Process Name contain in B.* | | | |

Table 17 Kill Process

#### <IT> Get real-time Process

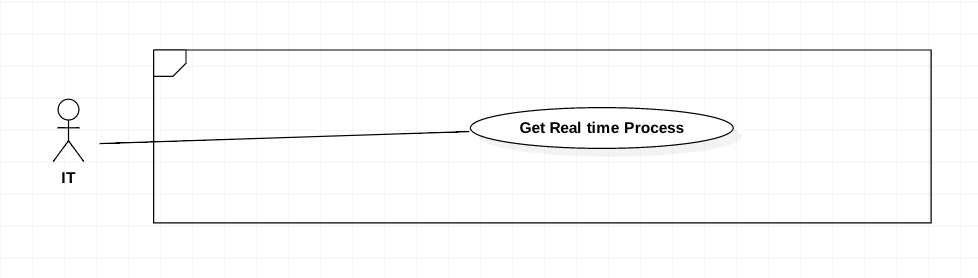


Figure 13 <IT> Get real-time Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.06** | | | |
| **Use Case No.** | UC\_IT06 | **Use Case Version** | 2.0 |
| **Use Case Name** | Get real-time 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:*   * *Process Name* * *Application Name* * *Process Path* |   **Business Rules:**   * *When student running the ExamTool, IT Managers can get all processes running on that computer. Process information contains:*   + *Process Name*   + *Application Name (Application run that process)*   + *Process Path (Path of application running that process)* | | | |

Table 18 <IT> GET REALTIME PROCESS

#### <IT> Search real-time Process

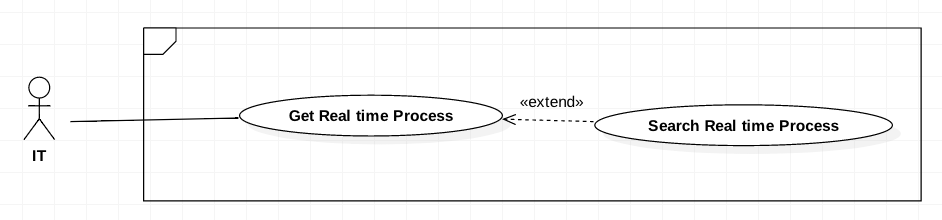


Figure 14 <IT> Search real-time Process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.07** | | | |
| **Use Case No.** | UC\_IT07 | **Use Case Version** | 2.0 |
| **Use Case Name** | Search real-time 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:*   * *Process Name* * *Application Name* * *Process Path* | | *2* | *IT input search value* | *The system matching searching value with process information:*   * *Process Name* * *Application Name* * *Process Path* |   **Relationships:**   * *Get real-time Processes*   **Business Rules:**   * *When student running the ExamTool, IT Managers can get all processes running on that computer (A). Process information contains:*   + *Process Name*   + *Application Name (Application run that process)*   + *Process Path (Path of application running that process)* * Searching value inputted by user (B) * Filter A which Process Name, Application Name, Process Path contain B | | | |

Table 19 <IT> SEARCH REALTIME PROCESS

#### <IT> Get Student Screen

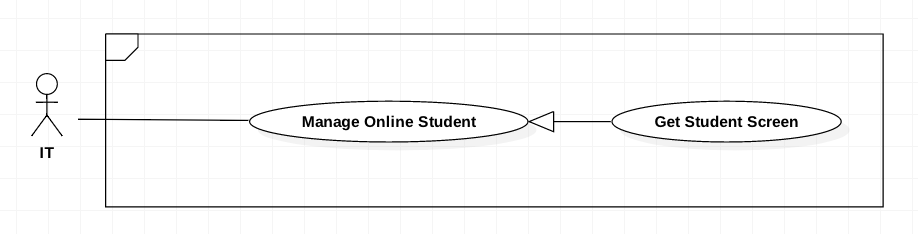


Figure 15 <IT> Get Student Screen

|  |  |  |  |
| --- | --- | --- | --- |
| **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* * *Connection ID* | | *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, User can get the list of those student who is using ExamTool (A).* * *Base on A, User choose student which user want to get screen (B). B’s information contains:*   *Student Name*  *Student Connection Id*   * *Call Get screen command to student application which have B’s Connection Id.* * *ExamTool capture continuous image and send back to Management page.* | | | |

Table 20 <IT> GET STUDENT SCREEN

#### <IT> Submit Student Test Exam

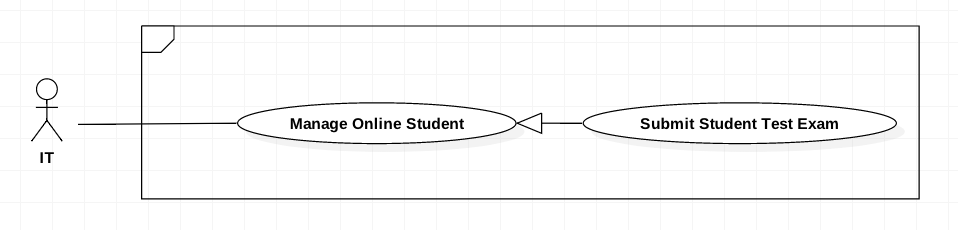


Figure 16 <IT> Submit Student Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **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 lost exam result when student client is crashed or get error in submitting 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* * *Connection ID* | | *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 connection ID 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 successes.* | *Show error message* |   **Business Rules:**   * *After students logged into the ExamTool, User can get the list of those student who is using ExamTool (A).* * *Base on A, User choose student which user want to get screen (B). B’s information contains:*   *Student Name*  *Student Connection Id*   * *Call submit student test exam command to student application which have B’s Connection Id.* * *Get hashed last test exam which saved local by student application (C). C’s information contains:*    + *StudentId*   + *ExamCode*   + *CourseCode* * *Parse C (D) and validate it format. If D valid format, check studentId, ExamCode, CourseCode to confirm is this test exam taken by B and not submitted.* * *If all fine, Insert D to test table.* | | | |

Table 21 <IT> Submit Student Test Exam

#### <IT> Search Online Student

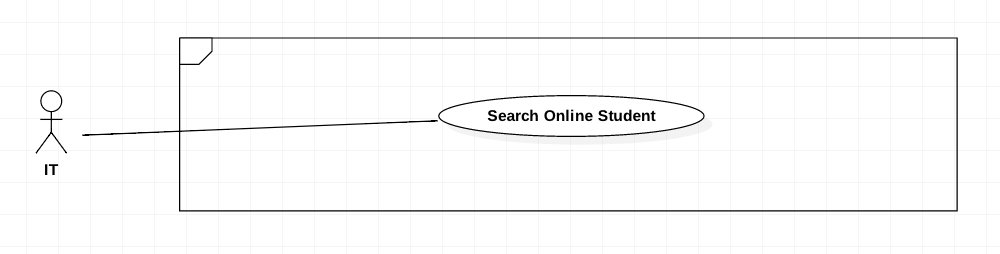


Figure 17 <IT> Search Online Student

|  |  |  |  |
| --- | --- | --- | --- |
| **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* * *Connection ID* | | *3* | *IT input search value* | *The system matching text between search value and online student information:*   * *Student User* * *Student Name* * *Connection ID* |   **Business Rules:**   * *After students logged into the ExamTool, User can get the list of those student who is using ExamTool (A). A’s information contains:*   + *Student Code*   + *Student Connection Id* * *Searching value inputted by user (B).* * *Filter A which A’s information contain search value.* | | | |

Table 22 <IT> SEARCH ONLINE STUDENT

#### <IT> Scan student by process

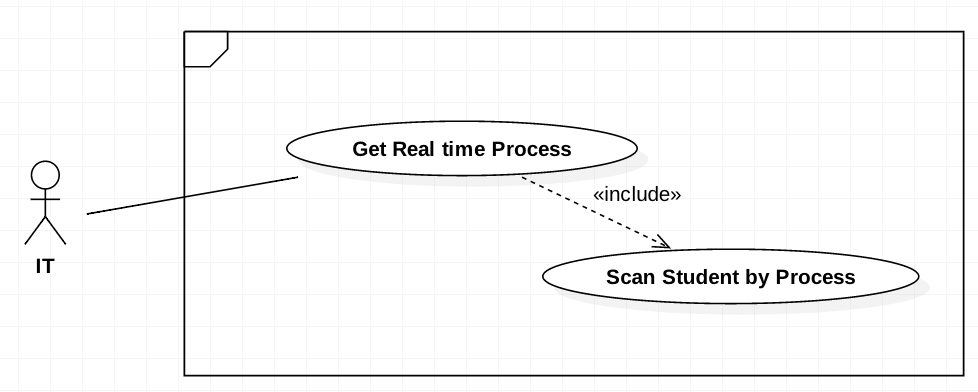


Figure 13 <IT> Scan student by process

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_IT.06** | | | |
| **Use Case No.** | UC\_IT06 | **Use Case Version** | 2.0 |
| **Use Case Name** | Scan student by process | | |
| **Author** | HieuCT | | |
| **Date** | 19/11/2018 | **Priority** | Normal |
| **Actor:**   * *IT*   **Summary:**   * *This use case allows user to get who running selected process*   **Goal:**   * *IT can scan who running selected process for preventing cheating.*   **Triggers:**   * *IT send Scan student by process command.*   **Preconditions:**   * *ExamTool must be running in student computer.*   **Post Conditions:**   * *Success: Get list student use selected process.* * *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 real time with information:*   * *Process Name* * *Application Name* * *Process Path* * *Student Connection Id* | | *2* |  | *The system will matching list Student in list online student with list process information. Display list student with information*   * *Student Code* * *Student Connection Id* |   **Business Rules:**   * *When student running the ExamTool, IT Managers can get all processes running on that computer (A). Process information contains:*   + *Process Name*   + *Application Name (Application run that process)*   + *Process Path (Path of application running that process)*   + *Student Connection Id of this process* * *Get list selected process by user (B)* * *Get all Student Connection Id and deduplicate B by Student Connection Id (C).* * *Get All Student who using ExamTool (D)* * *Filter D which D’s Student Connection Id contain in C.* | | | |

Table 23 <IT> Scan student by process

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

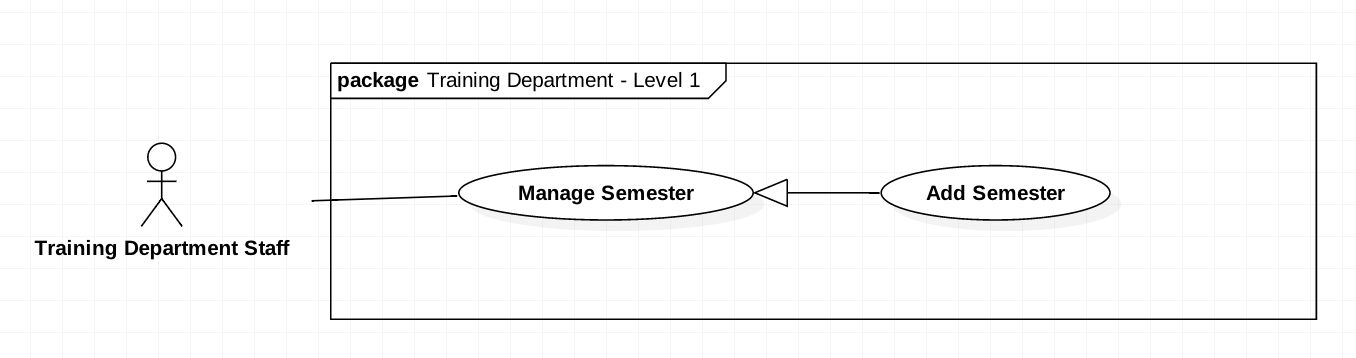


Figure 18 <Training Department Staff> Add new Semester

|  |  |  |  |
| --- | --- | --- | --- |
| **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: N/A**  **Business Rules:**   * + *Get the user input for a new semester(A)*   + *Get all semesters available in system(B)*   + *If A not existed in B, then insert A to system.* | | | |

Table 24 <TRAINING DEPARTMENT STAFF> ADD NEW SEMESTER

#### <Leader> Add chapter

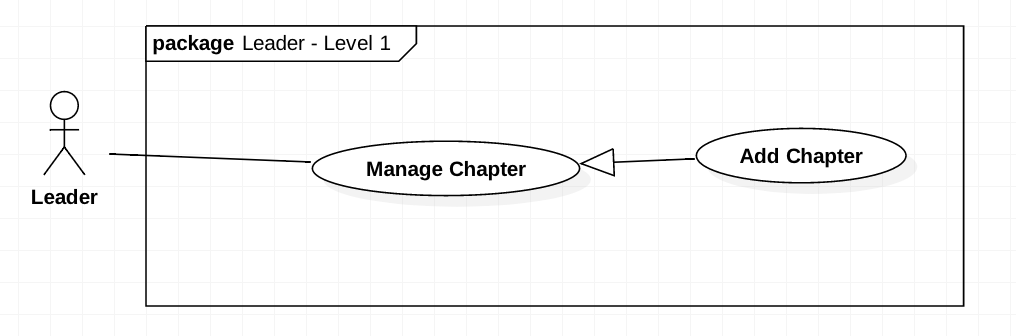


Figure 22 Add chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE-UC\_ Leader.151** | | | |
| **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: N/A**  **Business Rules:**   * + - *Get the user input for a new chapter(A).*     - *Get all chapters available base on course code in system(B).*     - *If A not existed in B, then insert A to system.* | | | |

Table 25 Add chapter

#### <Leader> Delete chapter

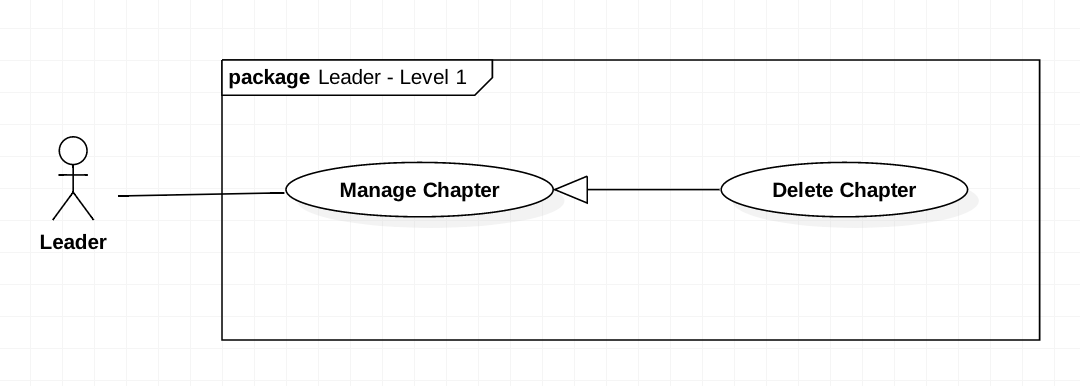


Figure 23 Delete chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **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*   **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: N/A**  **Business Rules:**   * *Get the user input for a chapter(A).* * *Get all chapters available base on course code in system(B).* * *If A existed in B, then remove A from system.* | | | |

Table 26 Delete chapter

#### <Leader> Edit chapter

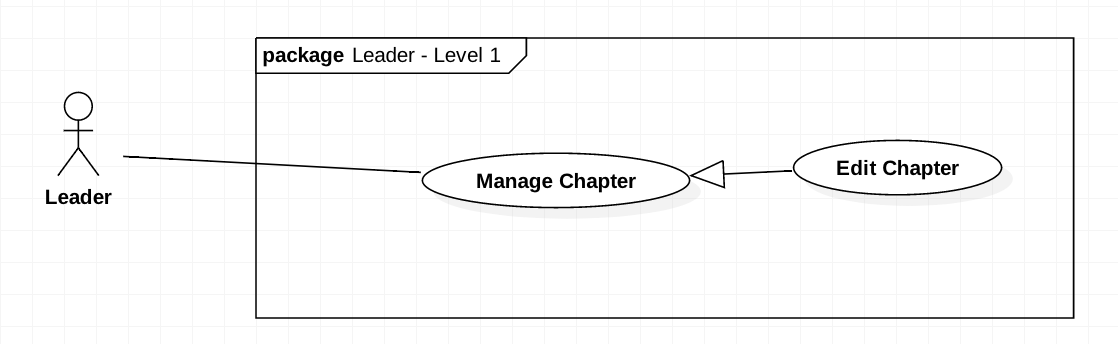


Figure 24 <Leader> Edit chapter

|  |  |  |  |
| --- | --- | --- | --- |
| **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 27 <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:**   * *Get question list base on course code (A).* * *Filter all questions has content contain search value in A (B).* * *System send back B to client.* | | | |

Table 29 <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 fill question form (A).*     - *A has format:*  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Content | Options | Mark | Level | Chapters | Learning outcomes |  * + - *Add A to question list.* | | | |

Table 30 <TEACHER> ADD QUESTION BY FORM

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

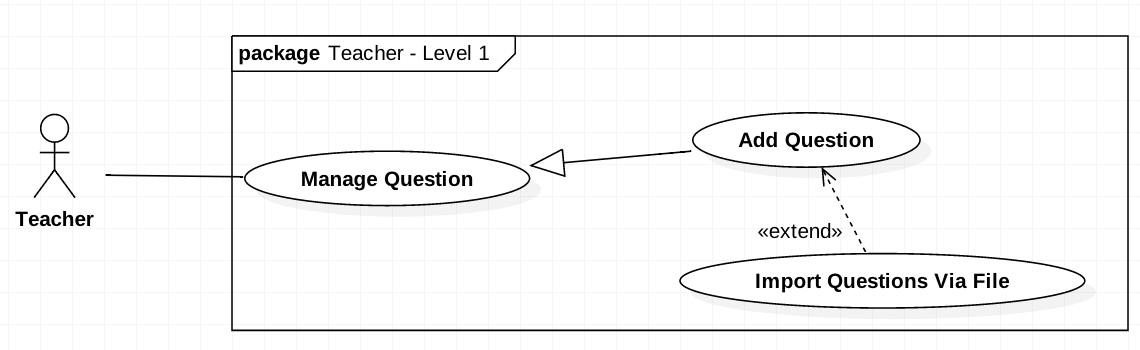


Figure 27 <Teacher> Add Question by Import File

|  |  |  |  |
| --- | --- | --- | --- |
| **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:**   * + - *Uploaded a file (A).*     - *Validate A by checking file type (.xml, .docx, .txt) and file format (GIFT format)(B)*     - *If B valid, parse B to question list.*     - *Question information has format:*  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Content | Options | Mark | Level | Chapters | Learning outcomes | | | | |

Table 31 <Teacher> Add Question by Import File

#### <Teacher> Remove Questions

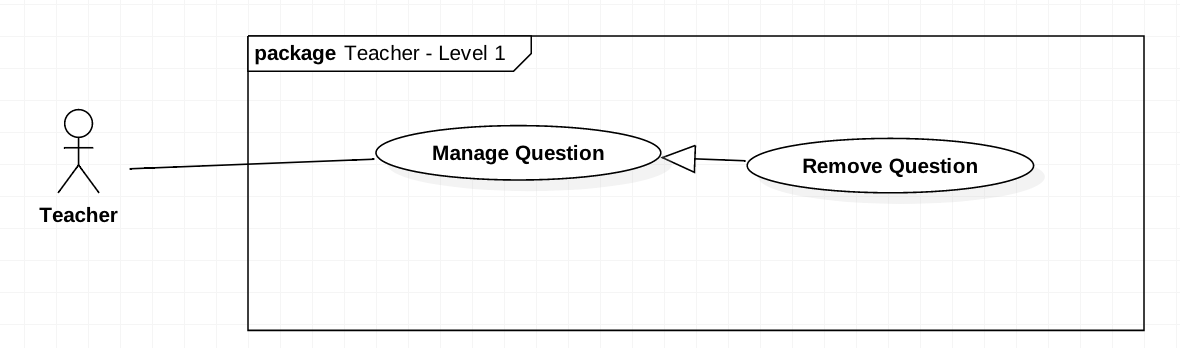


Figure 28 <Teacher> Remove Questions

|  |  |  |  |
| --- | --- | --- | --- |
| **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: N/A**  **Business Rules:**  *- Get all questions in system (A).*  *- Filter question in A base on question’s code (B).*  *- If B existed in A, remove B from system.* | | | |

Table 32 <Teacher> Remove Questions

#### <Teacher> Edit Questions

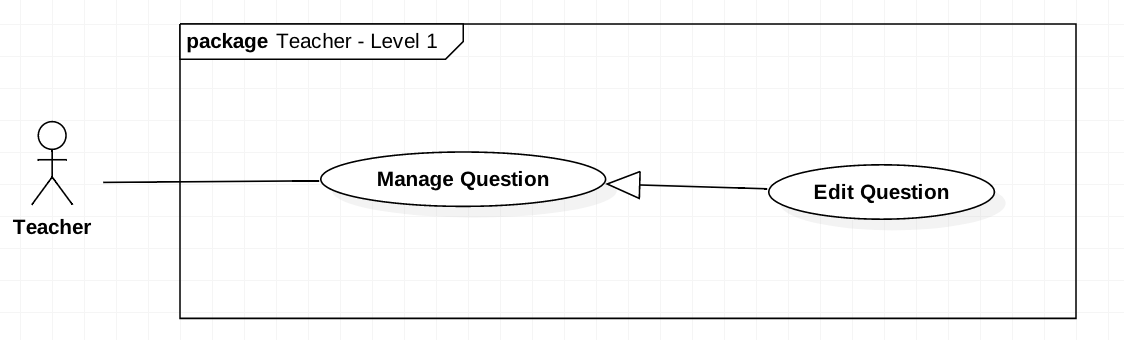


Figure 29 <Teacher> Edit Questions

|  |  |  |  |
| --- | --- | --- | --- |
| **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:**  *- Get the question base on question’s code (A)*  *- Update A by user input as format :*   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Content | Options | Mark | Learning outcomes | Chapters | Level |   *- Save A to system.* | | | |

Table 33 <Teacher> Edit Questions

#### <Teacher> Approve Test Exam

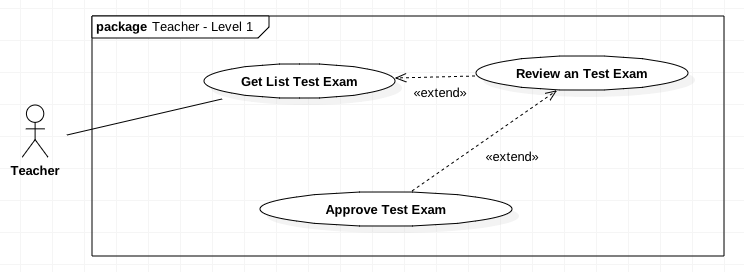


Figure 31 <Teacher> Approve Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **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:**   * *Get test exam base on code (A).*   + *If A’s status is “Pending”, teacher take an exam. After A finished, A’s status change to “Taken”.*   + *If A’s status is “Taken:*     - *If teacher update A, A’s status change to “Edited”.*     - *If teacher approve. A’s status change to “Approved”.*     - *If teacher reject, A’s status change to “Rejected”.*   + *If A’s status is “Edited”, teacher available to retake and A’s status change to “Pending”.* | | | |

Table 34 <Teacher> Approve Test Exam

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

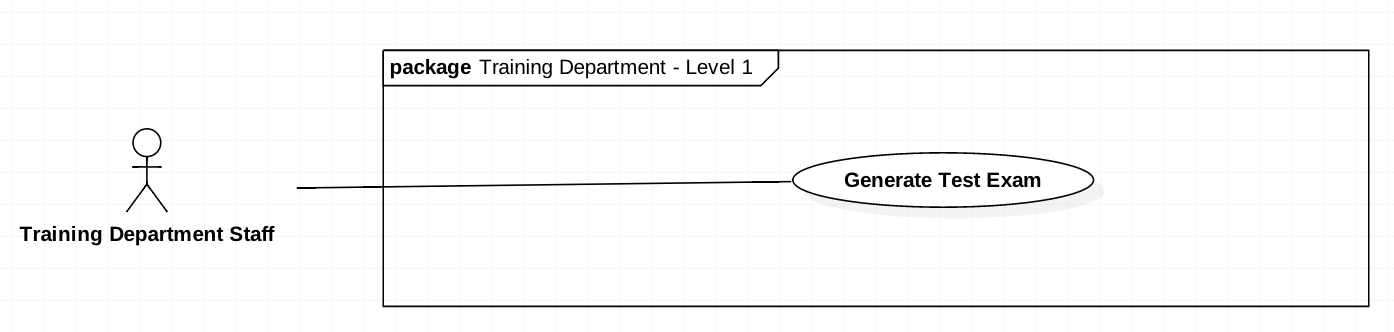


Figure 32 <Teacher> generate Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **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. *Get user’s input as format (A):* * *CourseCode* * *Chapters* * *Questions* * *LearningOutcomes* * *NumberOfTest* * *NumberOfQuestions* * *Duration* * *Type*   1. *If A’s type is “Manufacturing”, system create test exam base on A’s questions.*   2. *If A’s type is “LearningOutcomes”, system generate test exam base on A’s LearningOutcomes.*   3. *If A’s type is “Chapters”, system generate test exam base on A’s Chapters.* | | | |

Table 35 <Training department staff> Generate Test Exam

#### <Training Department Staff> Assign Approver

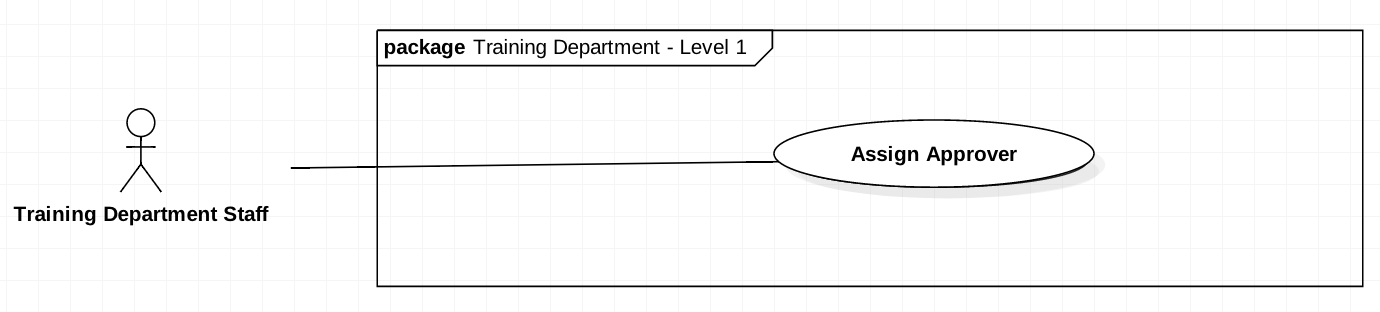


Figure 33 < Training Department Staff > Approve Test Exam

|  |  |  |  |
| --- | --- | --- | --- |
| **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. *Get test exam base on code (A).*   2. *Update A by user’s input as format:*  |  |  |  | | --- | --- | --- | | Approver | StartDate | EndDate |  * 1. *System save A.* | | | |

Table 36 <Training department staff> Assign Approver

## Software System Attribute

### Usability

* + Font size: 8px – 30px.
  1. Color: Light Red, Light Green, Blue, White, Light Gray
  2. 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

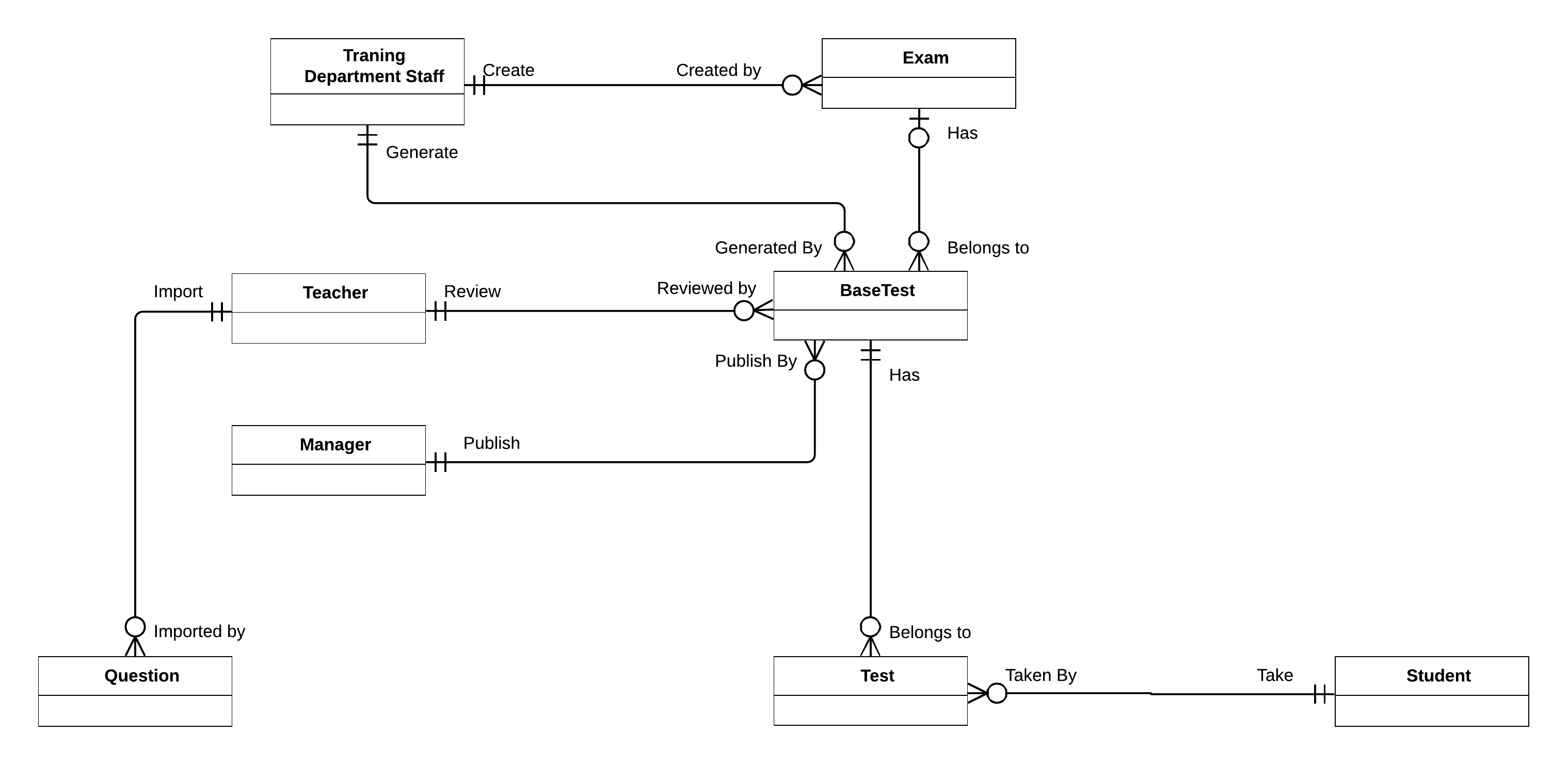


Figure 34 Conceptual Diagram

|  |  |
| --- | --- |
| **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 |
| BaseTest | Contain test exam information |
| Teacher | The people who import question and approve test exam |
| Manager | The people who publish test exam |
| Question | Contain questions information |
| Test | Contain student’s test information |
| Student | Contain Student’s information |

Table 37 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

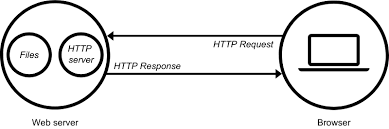


Figure 35 System Architecture Design

### 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.

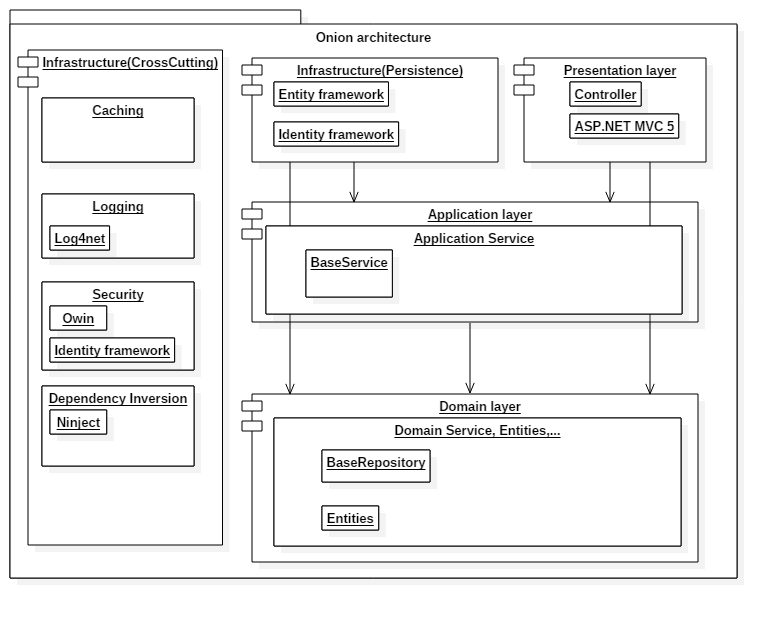


Figure 36 Architecture Overview

|  |  |  |
| --- | --- | --- |
| **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 38 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.

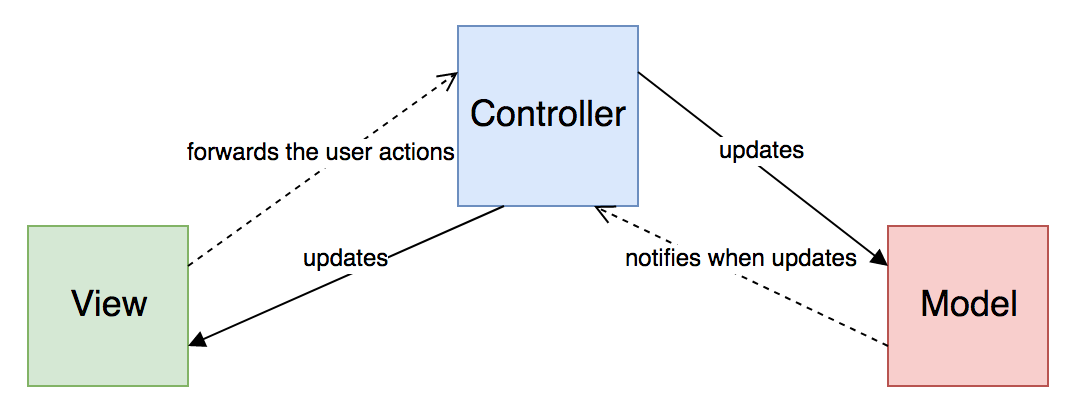


Figure 37 Web Admin MVC Architecture

(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

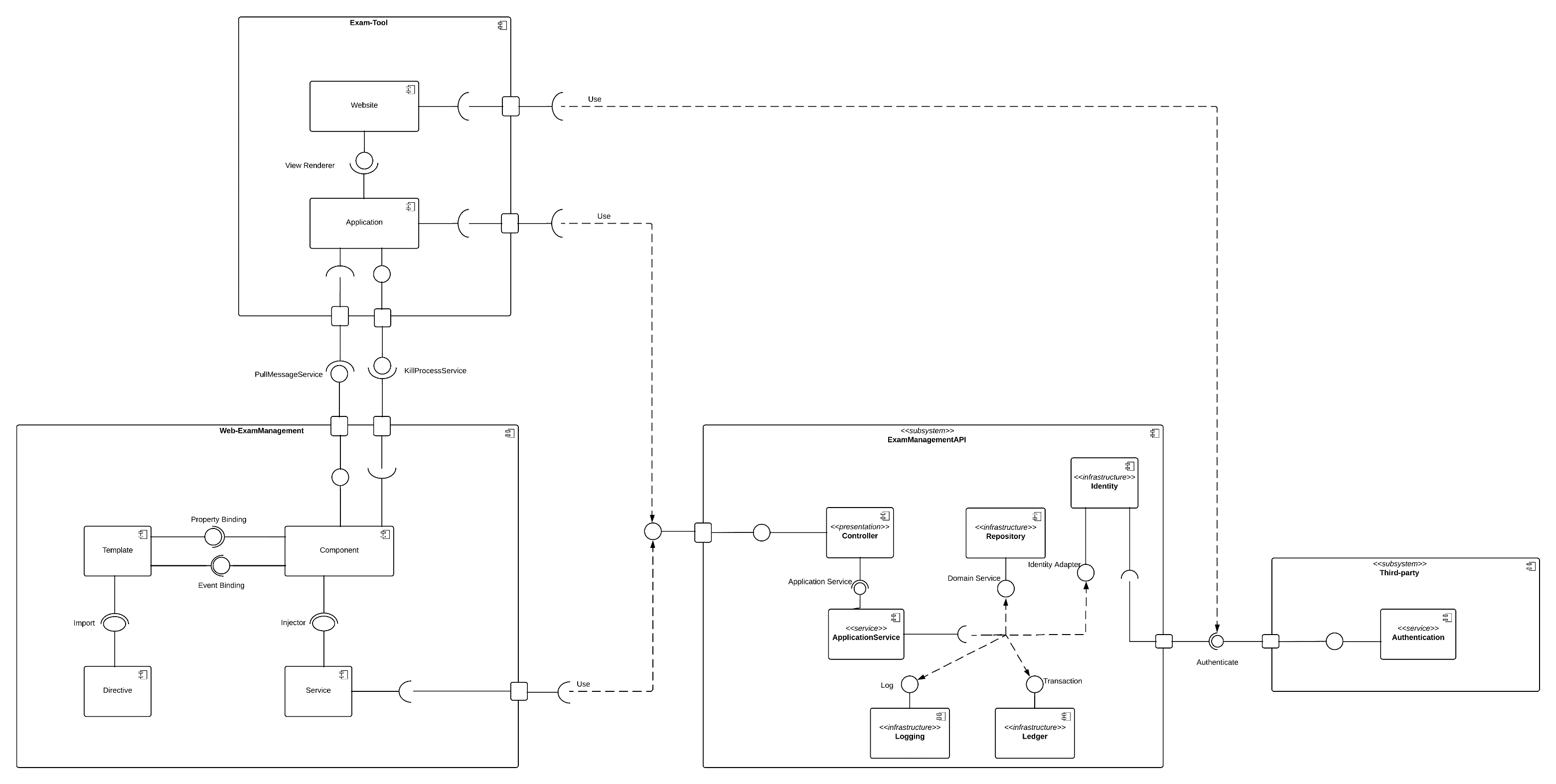


Figure 38 Component Diagram

|  |  |
| --- | --- |
| **Entity Name** | **Description** |
| Exam-tool | This subsystem provide tool help Student can take a test exam and through this tool IT can manage process on student computer. This subsystem include 2 main component:   * 1. Website take a test exam   2. Application |
| Website | Provide interface for student take a test exam |
| Application | Provide tool for IT can collect process on student’s computer and send to IT management website. It can screenshot student screen. |
| Web-Exam Management | This web build for manage and build by angular with angular structure. This website use API provide by ExamManagementAPI subsystem |
| Authenticate service | This service provide by FPT University. (Login, Get User info…) |
| ExamManagementAPI | Provide API for System. |

Table 39 Component diagram dictionary

## Detailed Description

### Class Diagram

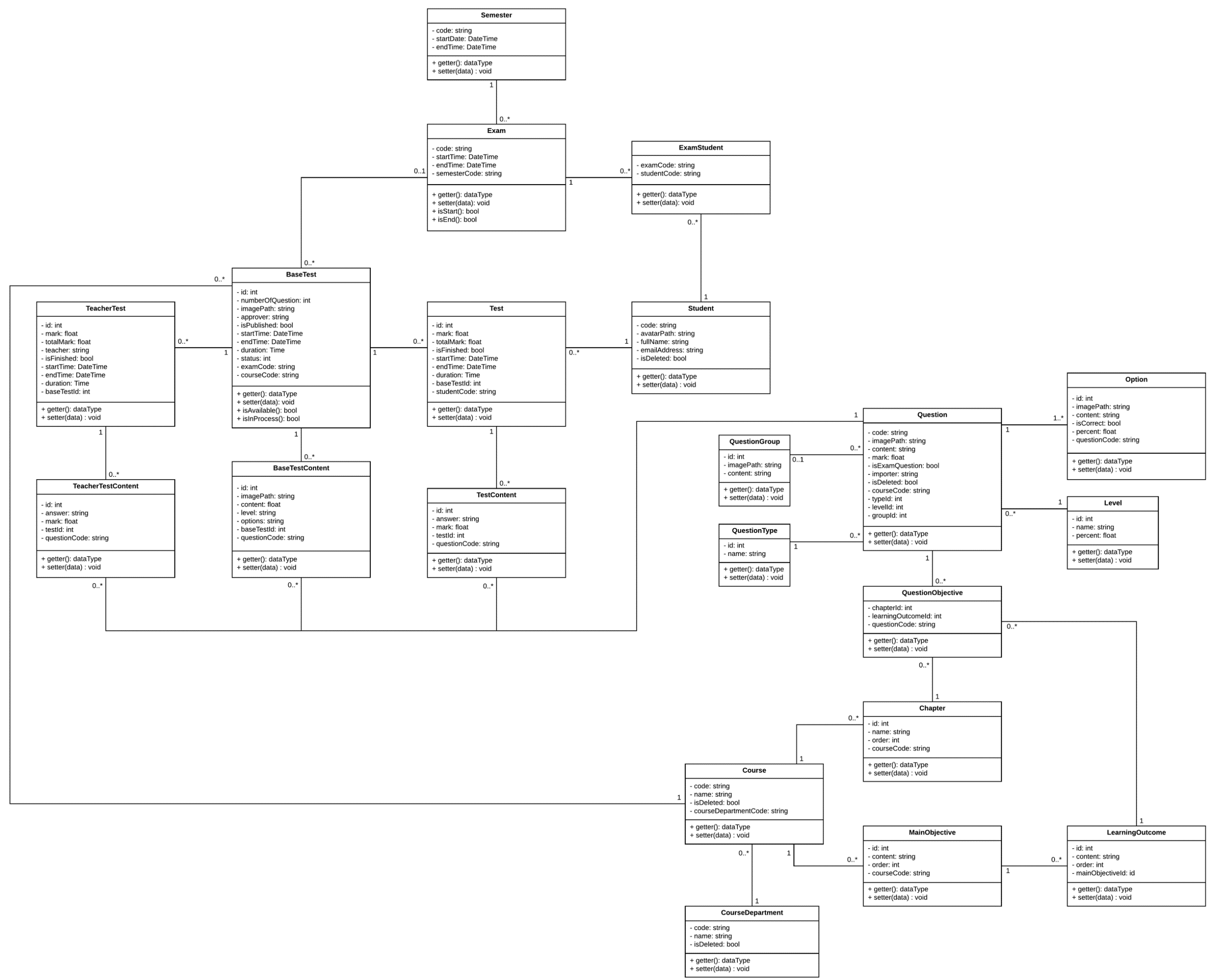


Figure 39 Class diagram

|  |  |  |
| --- | --- | --- |
| **Entity Data dictionary: describe all content of all entities** | | |
| **Class Name** | **Mapping column with conceptual diagram** | **Description** |
| BaseTest | BaseTest | 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 | Test | 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 | Question | 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 40 Class diagram Description

### Class Diagram Explanation

#### Semester

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 41 Class Semester Explanation

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 42 Class Semester method explanation

#### Exam

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 44 Class exam method explanation

#### ExamStudent

**Attribute**

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

Table 45 Class ExamStudent Explanation

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 46 Class ExamStudent method explanation

#### Student

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 47 CLASS STUDENT EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 48 Class student method explanation

#### BaseTest

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 49 CLASS BASETEST EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 50 Class basetest method explanation

#### BaseTestContent

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 51 CLASS BASETESTCONTENT EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 52 Class basetestcontent method explanation

#### Test

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 53 CLASS TEST EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 54 Class test method explanation

#### TestContent

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 55 CLASS TESTCONTENT EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 56 Class testcontent method explanation

#### Question

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 57 CLASS QUESTION EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 58 Class question method explanation

#### Option

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |
| Percent | double | Private | Percent will Minus if choice wrong |

Table 59 CLASS OPTION EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 60 Class option method explanation

#### Level

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 61 CLASS level EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 62 Class level method explanation

#### QuestionType

**Attribute**

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

Table 63 CLASS QUESTIONTYPE EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 64 Class questiontype method explanation

#### QuestionGroup

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 65 CLASS QUESTIONGROUP EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 66 Class questiongroup method explanation

#### QuestionObjective

**Attribute**

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

Table 67 CLASS QUESTIONOBJECTIVE EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 68 Class questionobjective method explanation

#### Chapter

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 69 CLASS CHAPTER EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 70 Class chapter method explanation

#### Course

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 71 CLASS COURSE EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 72 Class course method explanation

#### MainObjective

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 73 CLASS MAINOBJECTIVE EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 74 Class mainobjective method explanation

#### LearningOutcome

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 75 CLASS LEARNINGOUTCOME EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 76 Class learningoutcome method explanation

#### TeacherTest

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 77 CLASS TeacherTest EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 78 Class teachertest method explanation

#### TeacherTestContent

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 79 CLASS TeacherTestContent EXPLANATION

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Return Type** | **Visibility** | **Description** |
| Getter | Attribute Type | private | Get attribute value |
| Setter | Void | private | Set attribute Value |

Table 80 Class teachertestcontent method explanation

### Interaction Diagram

#### Activity Diagram

##### Teacher Import Question

**Summary:** This diagram shows how Teacher import question

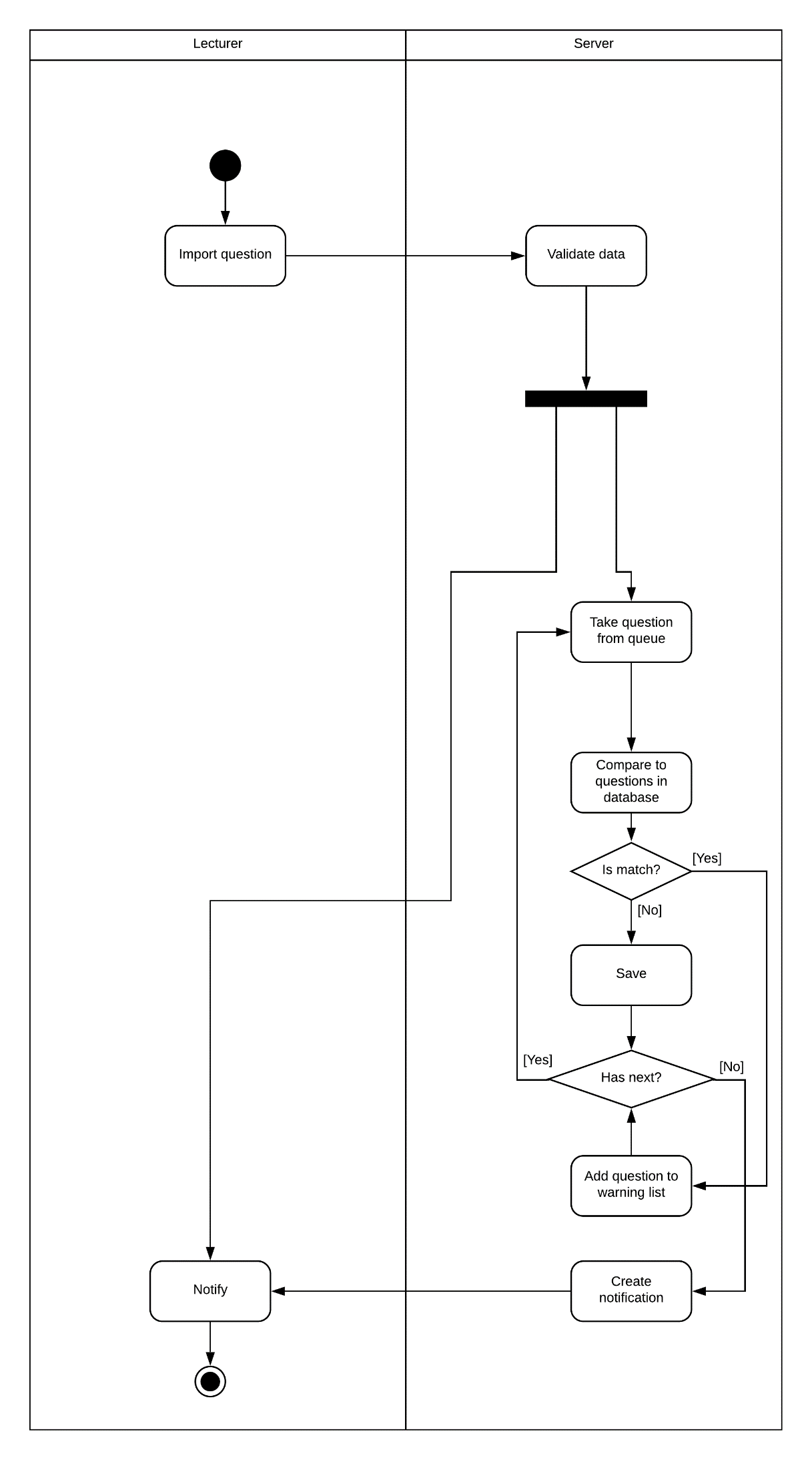


Figure 40 Activity diagram Teacher Import Question

##### Generate Test Exam

**Summary:** This diagram shows how Staff Create Test Exam

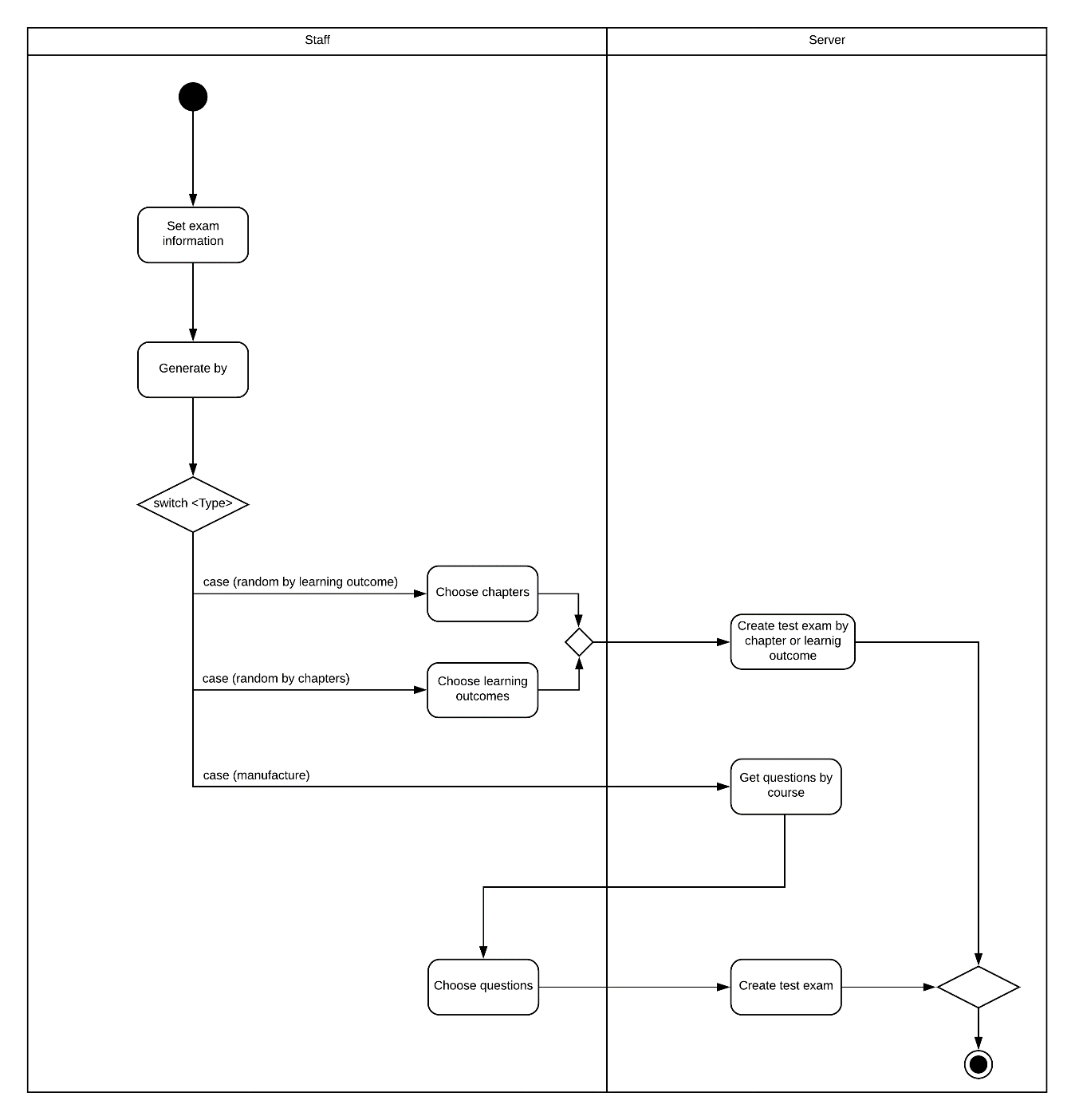


Figure 41 Activity diagram Training department staff generate Test Exam

##### Teacher Approve Test Exam

**Summary**: This diagram shows how Teacher approve test exam

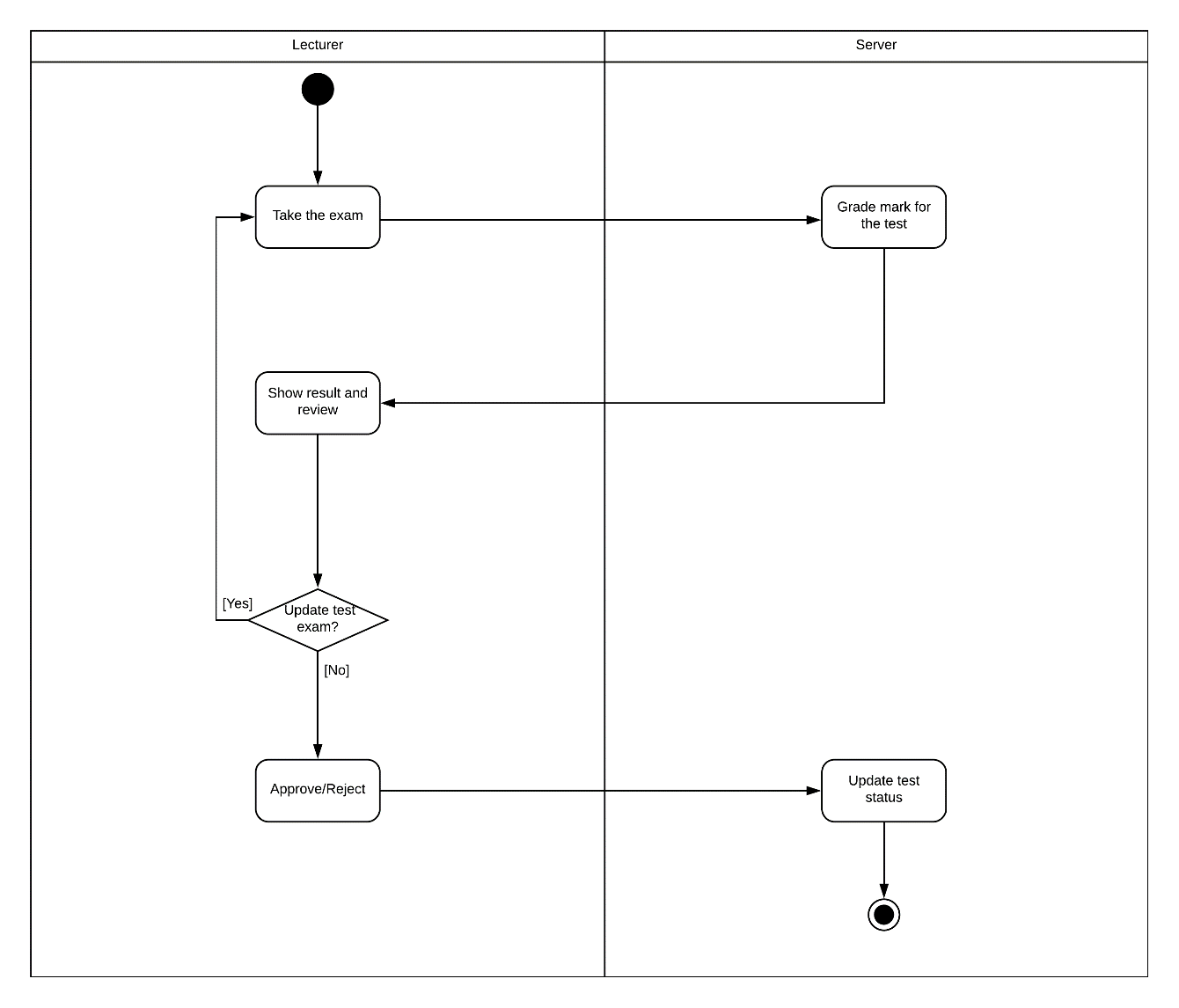


Figure 42 Activity diagram Approve Test Exam

##### Student Take a Test Exam

**Summary**: This diagram shows how Student Take a Test Exam

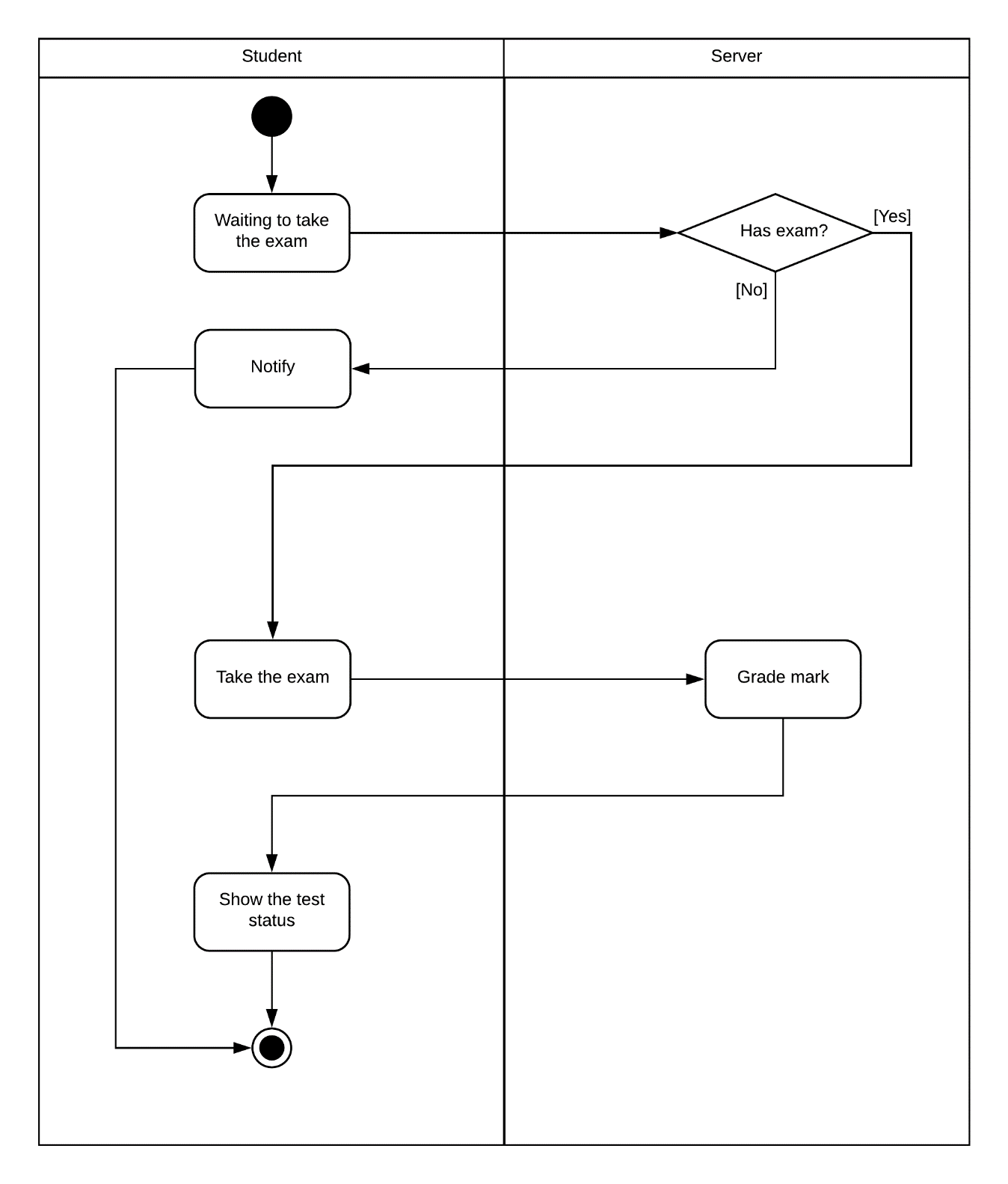


Figure 43 Activity diagram Student take a test exam

## Interface

### Manage Question Bank

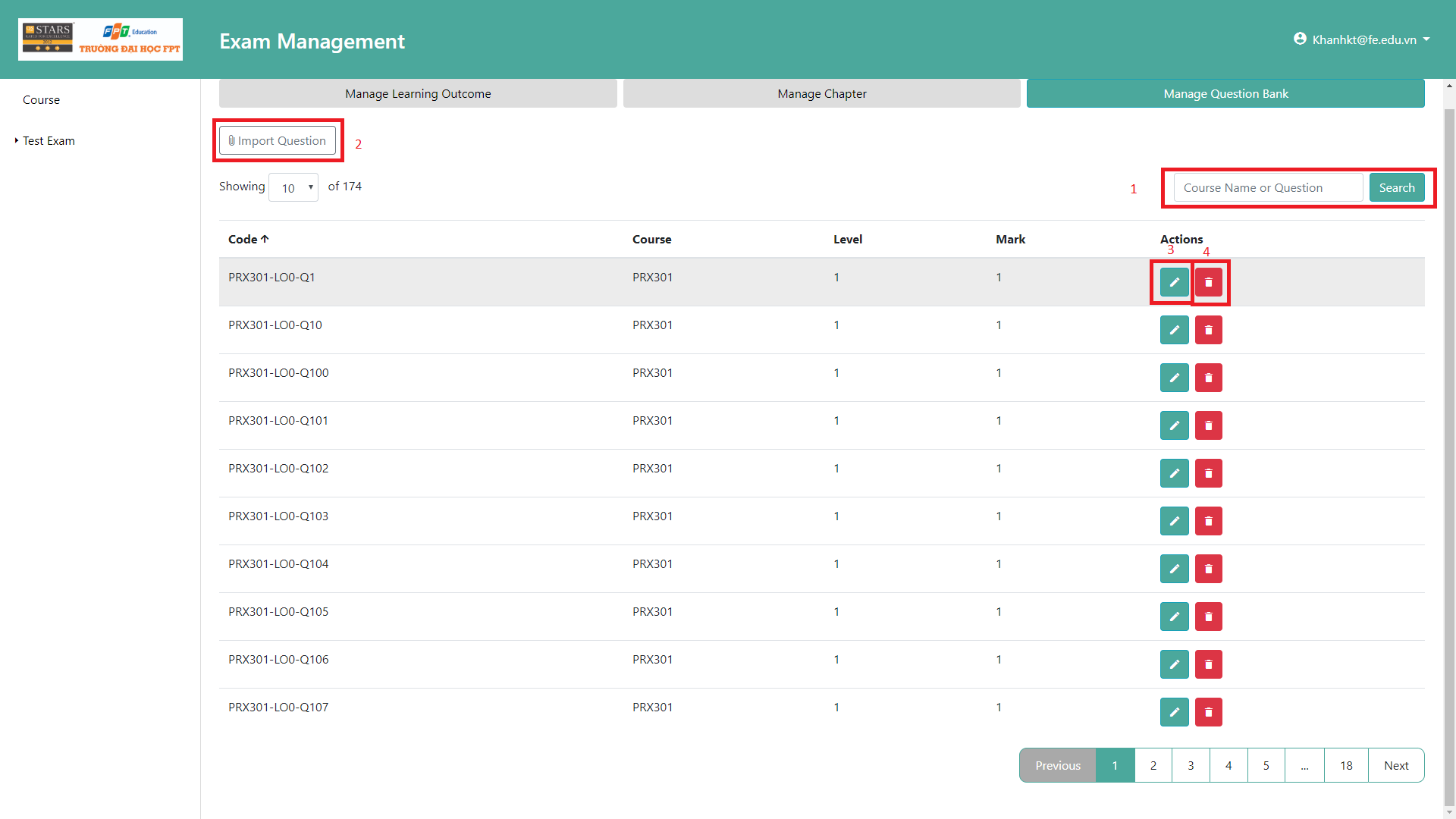


Figure 44 manage question bank

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | SearchText | Fill in question code or question content to filter question list | No | No | Text | String | 0-255 characters |

Table 81 manage question bank fields

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | RedirectToImportQuestion | Go to import question page | N/A | Move to import question page |
| 3 | EditQuestion | Edit current question | N/A | Show a popup with question detail |
| 4 | DeleteQuestion | Delete current question | N/A | Delete and refresh the question list |

Table 82 manage question bank buttons

### Import question

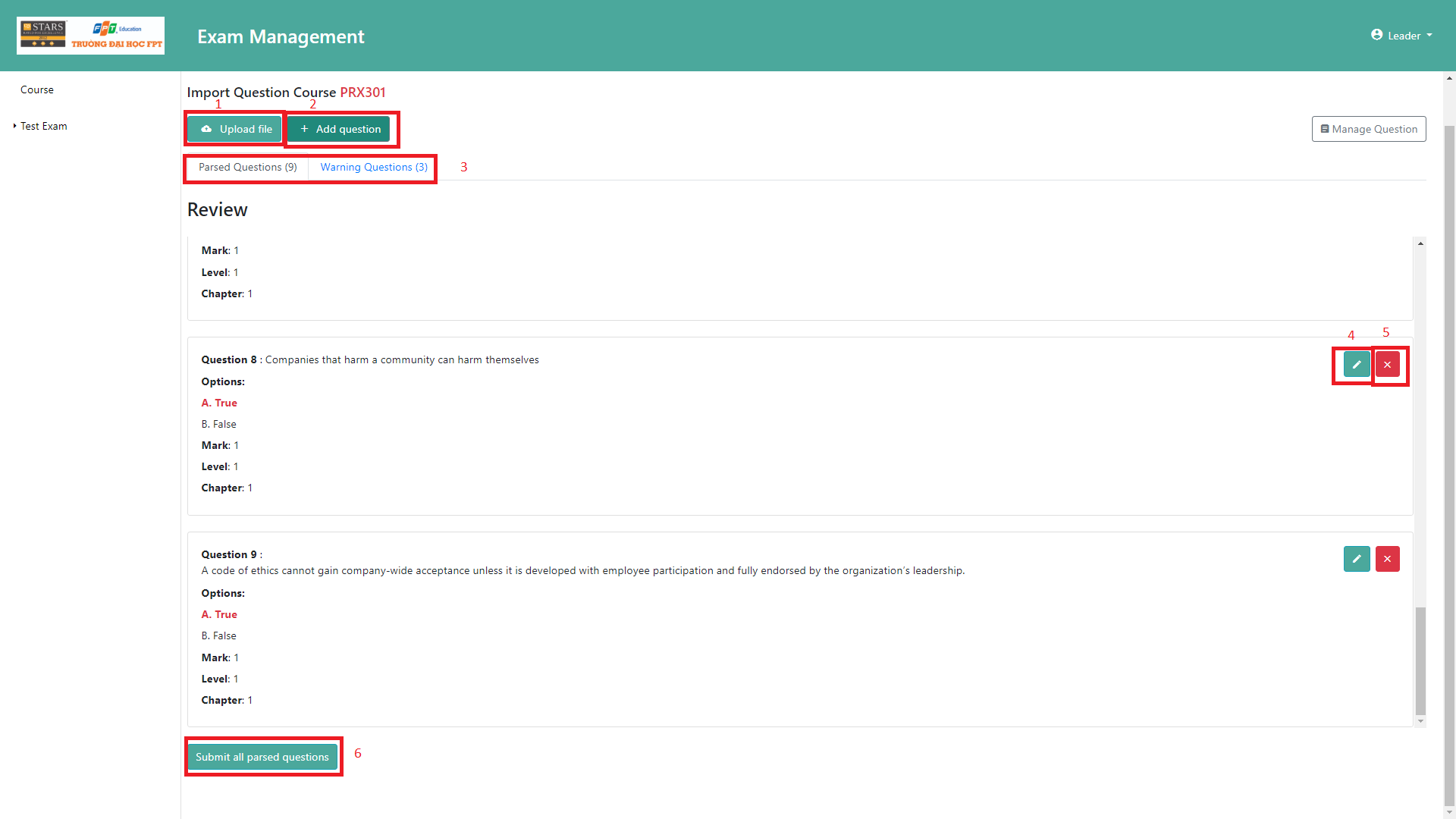


Figure 45 import question

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | UploadFile | Upload a file | File must be XML, docx or txt (Gift Format). | Parsed file to question list |
| 2 | AddQuestion | Add a question by using form | N/A | Show a popup |
| 3 | ChangeTab | Show all parsed questions successful or show all invalid questions | N/A | Show content of tab is chosen |
| 4 | EditQuestion | Edit current question | N/A | Show a popup with question detail |
| 5 | DeleteQuestion | Delete current question | N/A | Delete question in the question list |
| 6 | Submit | Send question list to server to matching | Question list cannot be emptied | Validate and start marching question. |

Table 83 import question buttons

### Matching question



Figure 46 matching question

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | ShowDetail | Show detail of current process | N/A | Show detail of process below the table |
| 2 | DeleteQuestion | Delete question from list | Current process must finish matching. | Delete question. |
| 3 | MoveNext/MovePrev | Drag to move next or move previous question in the list | Current process must finish matching | Show next or previous question. |
| 4 | Verify | Send question list to server | Current process must finish matching | Save the question list to question bank |

Table 84 matching question

### Create Test Exam

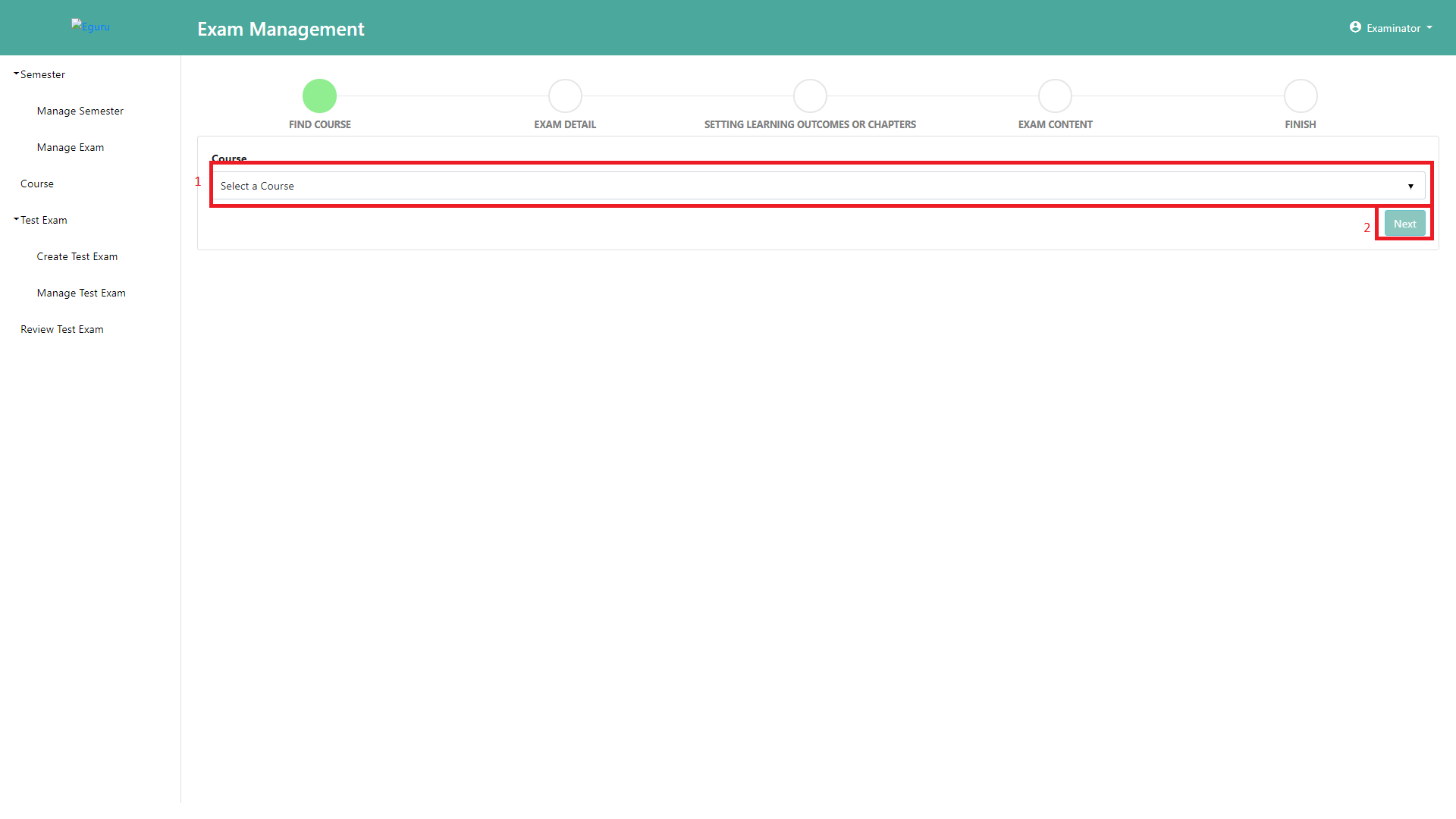


Figure 47 create test exam

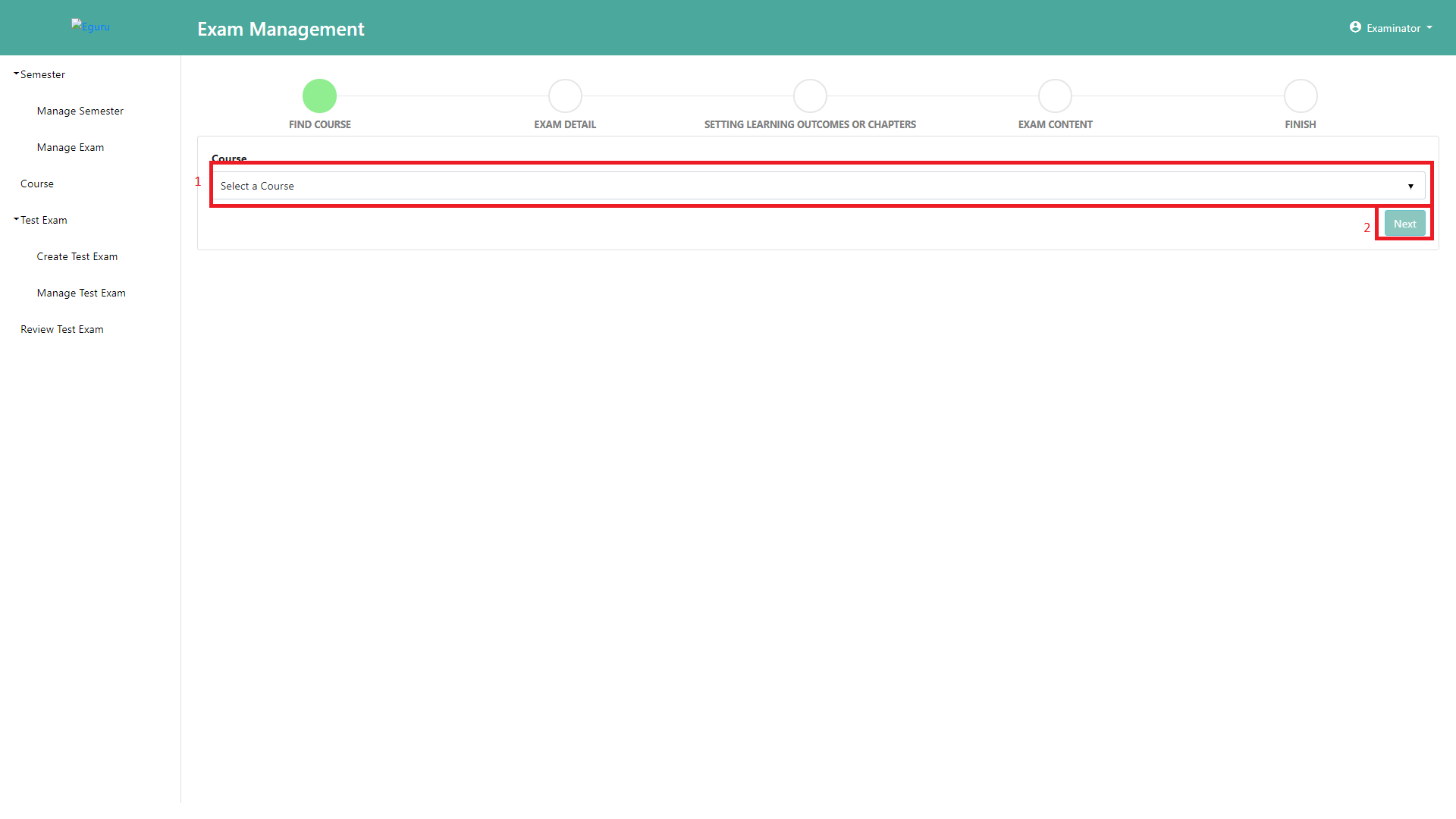


Figure 48 create test exam

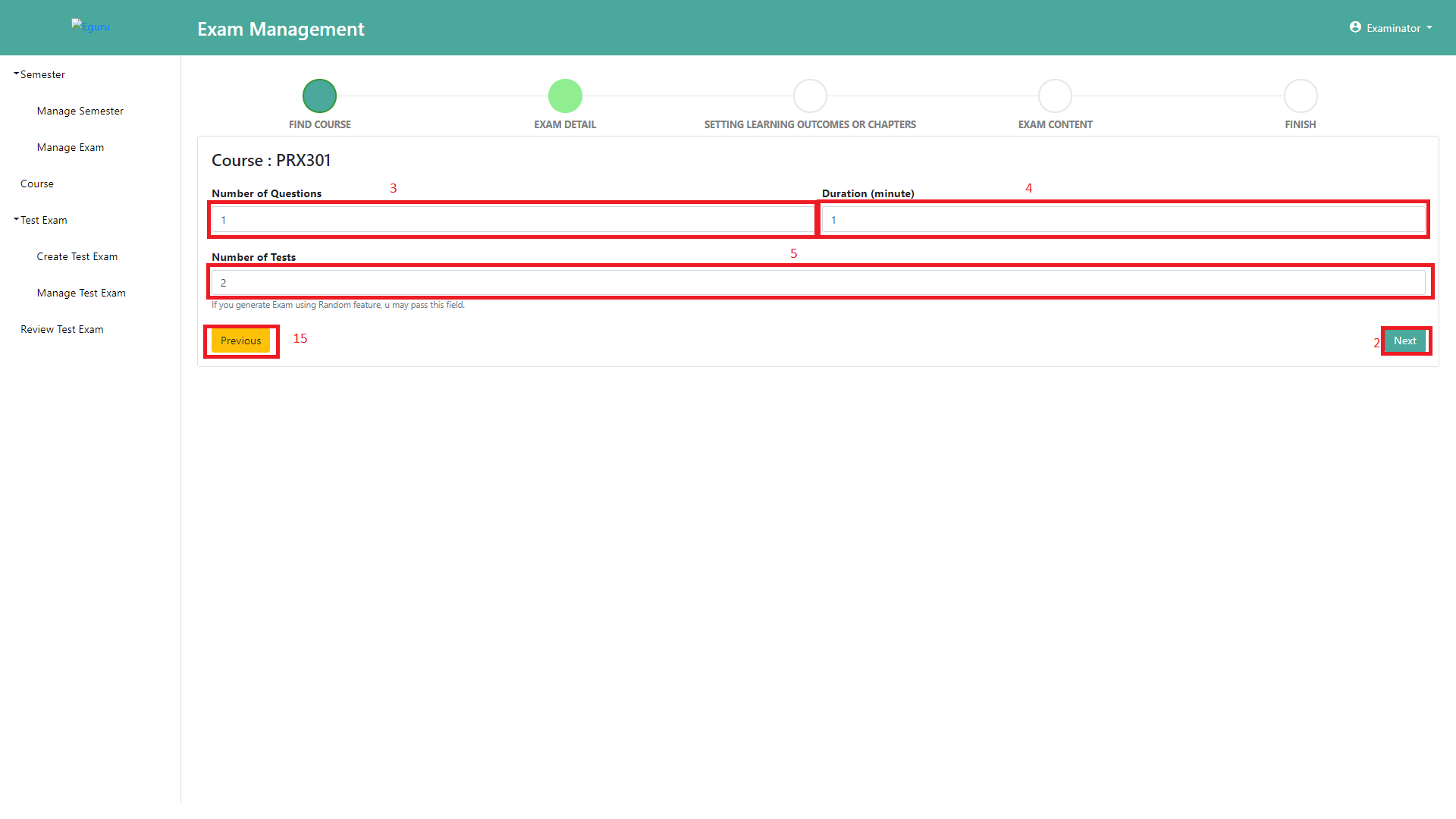


Figure 49 create test exam

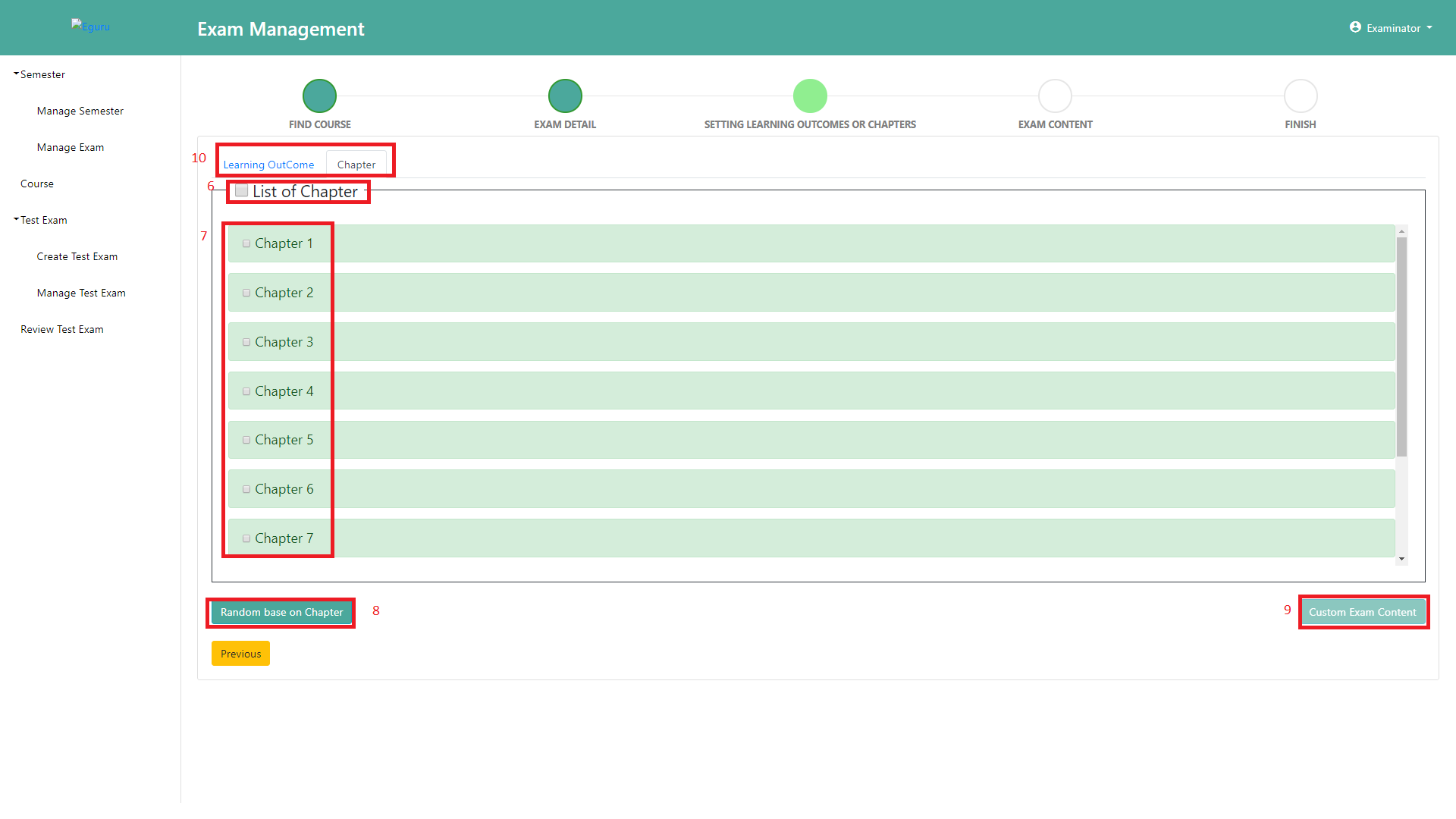


Figure 50 create test exam

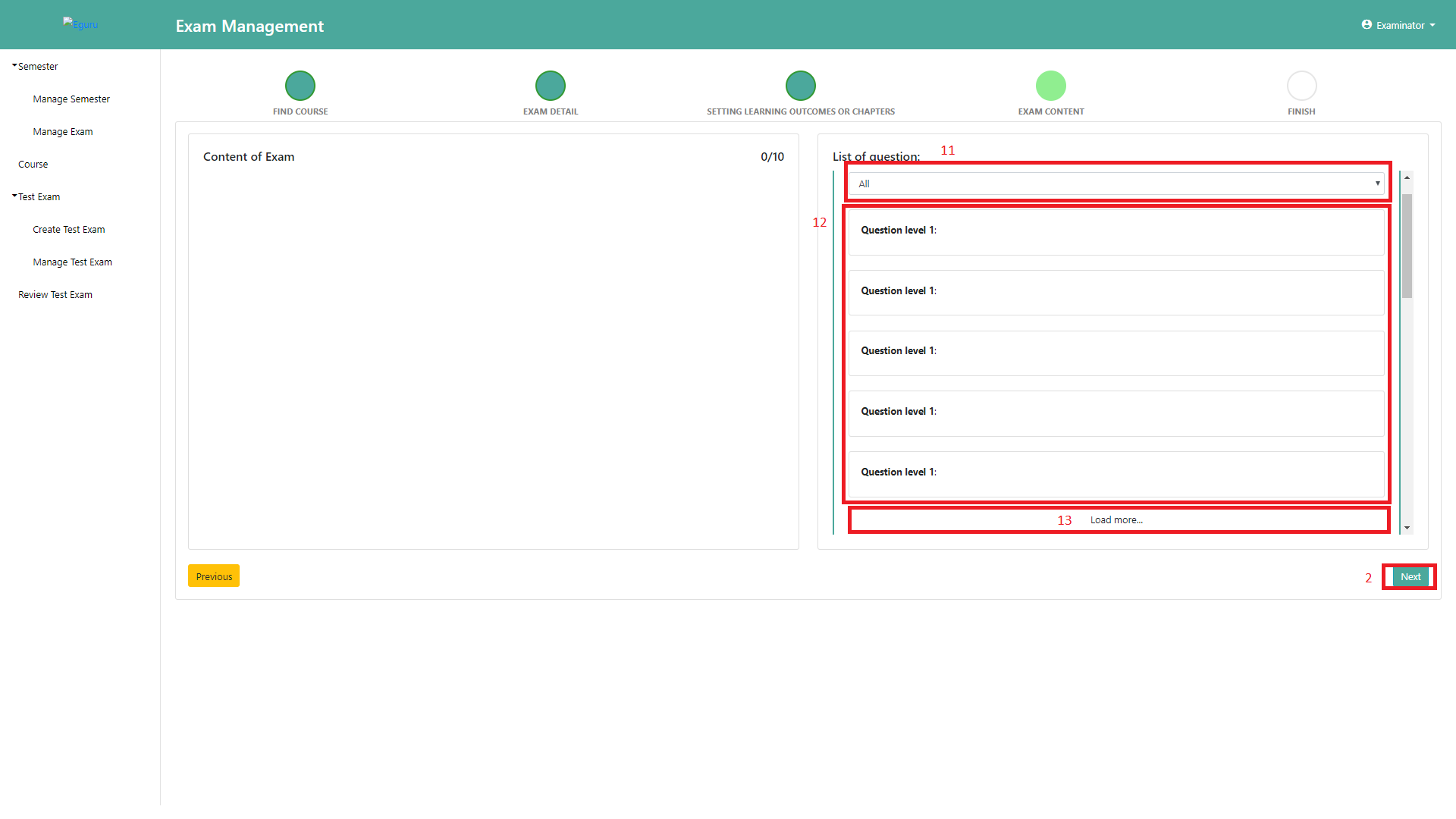


Figure 51 create test exam

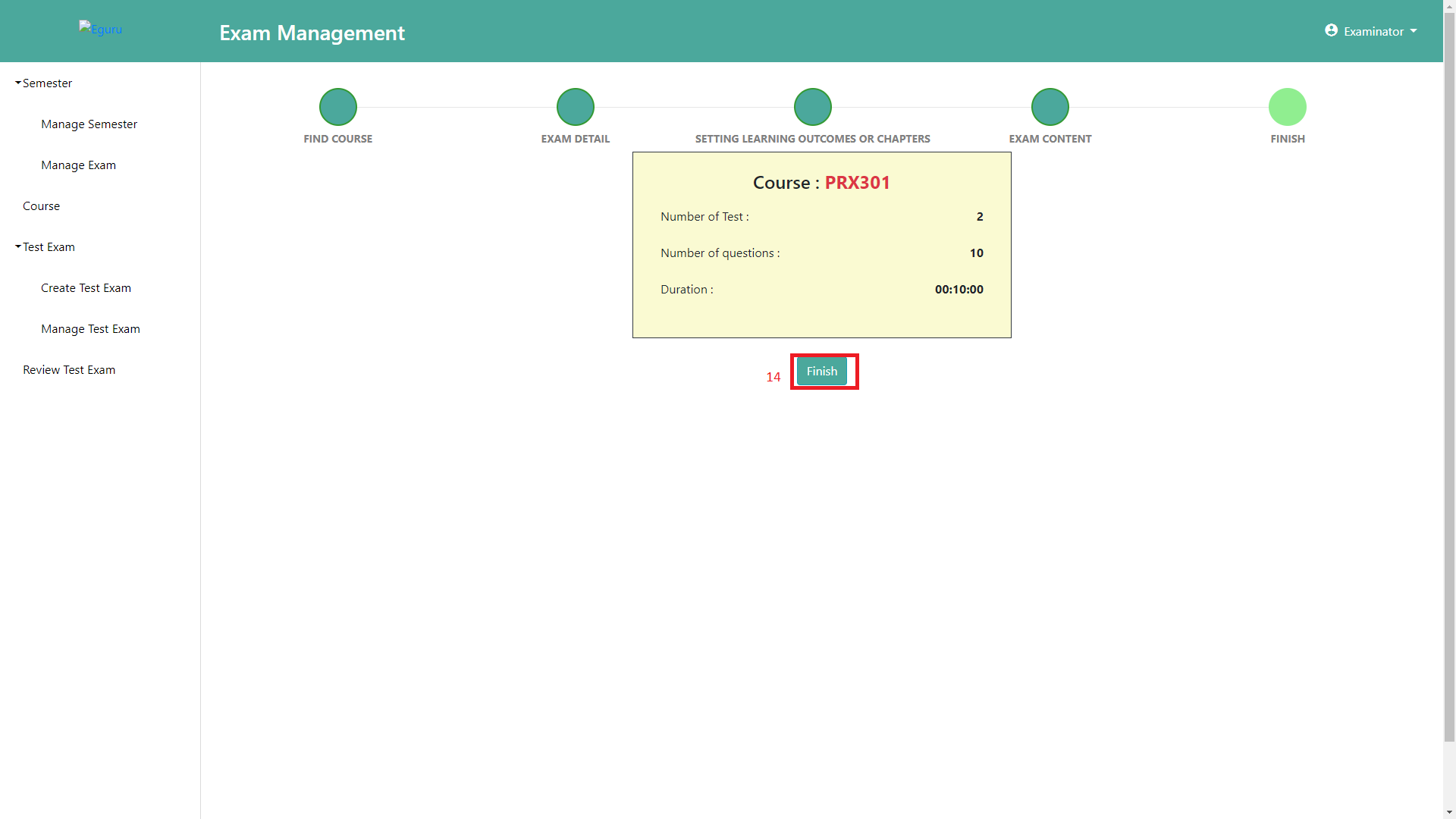


Figure 52 create test exam

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | CourseCode | Fill a code course | No | Yes | Select | String | 1 |
| 3 | NumberOfQuestions | Fill number of questions in the test exam | No | Yes | EditText | Integer | 1-100 |
| 4 | Duration | Fill duration of a test exam | No | Yes | EditText | Integer | 1-1439 |
| 5 | NumberOfTest | Fill number of test exam will be generated | No | Yes | EditText | Integer | 1-10 |
| 11 | Level | Fill in level to filter question list of a chapter/learning outcome | No | No | Select | Integer | 1 |

Table 85create test exam fields

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | NextStep | Move to next step | N/A | Move to next step |
| 6 | CheckAll | Select all chapters or learning outcome | No | Select all chapters or learning outcomes |
| 7 | Check | Select a chapter or learning outcome | No | Select a chapter or learning outcome |
| 8 | RandomTestExam | Random test base on NumberOfTest and chapter list or learning outcome list. | N/A | Show a popup with question detail |
| 9 | SetList | Delete current question | Chapters list or Learning Outcome list cannot be emptied | Delete and refresh the question list |
| 10 | ChangeTab | Show all chapters or learning outcomes | N/A | Show content of tab is chosen |
| 12 | SetQuestion | Drag to left column to add this question to test exam content | Test Exam Content is lower than NumberOfQuestions | Add question to test exam content |
| 13 | LoadMore | Load next page and merge to question list of current chapter or learning outcome. | N/A | Show more questions. |
| 14 | Finish | Move back to step one | N/A | Move back to step one |
| 15 | PreviousStep | Move to previous step | N/A | Move to previous step |

Table 86 create test exam buttons

### Test Exam List with “Pending” Status

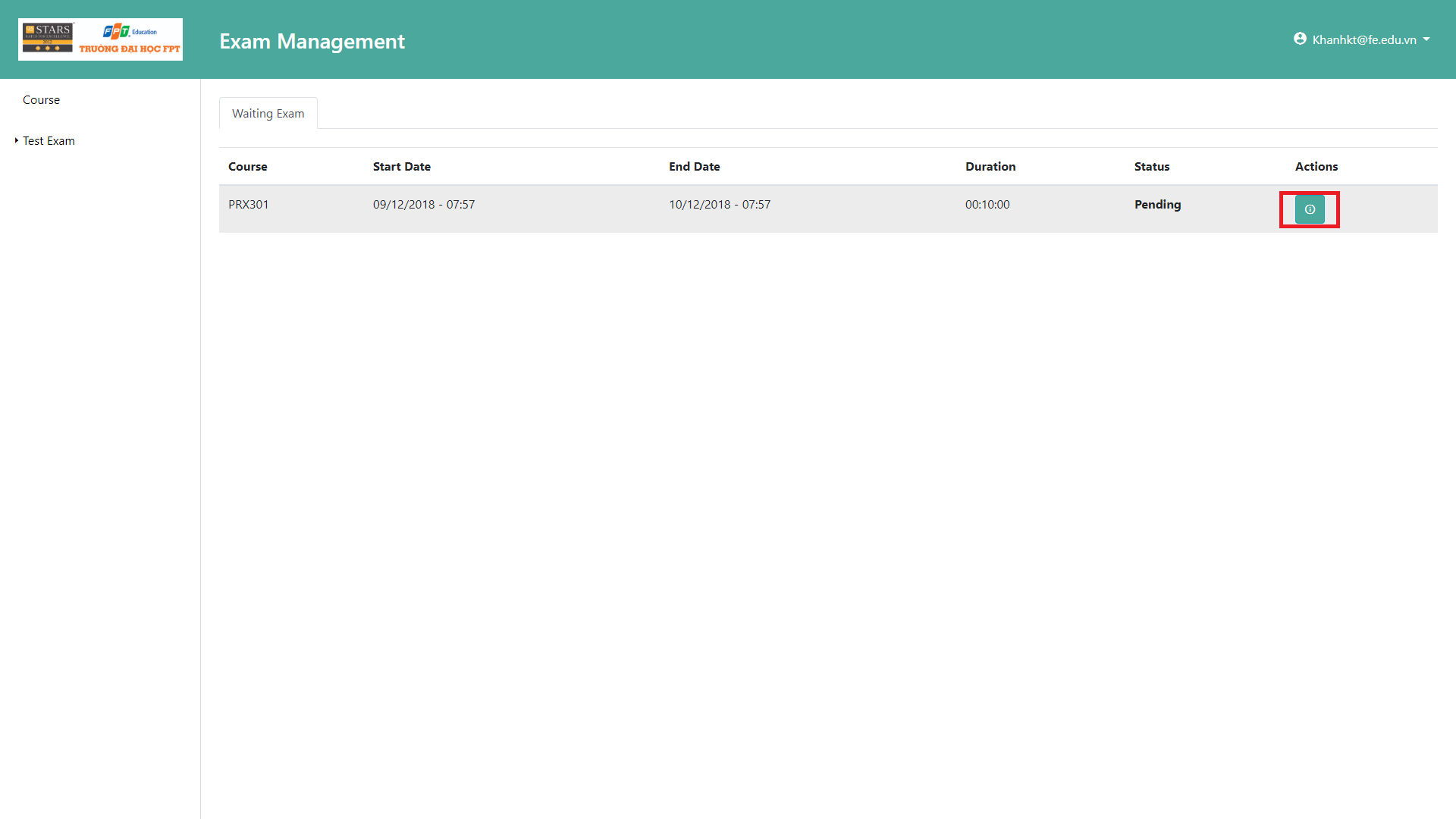


Figure 53 test exam list with "pending" status

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Detail | View detail of test exam | Start Date of test exam must greater than or equal current time.  End Date of test exam must lower than or equal current time. | Go to “Detail” page |

Table 87 test exam list with “pending” status buttons

### Take an Exam

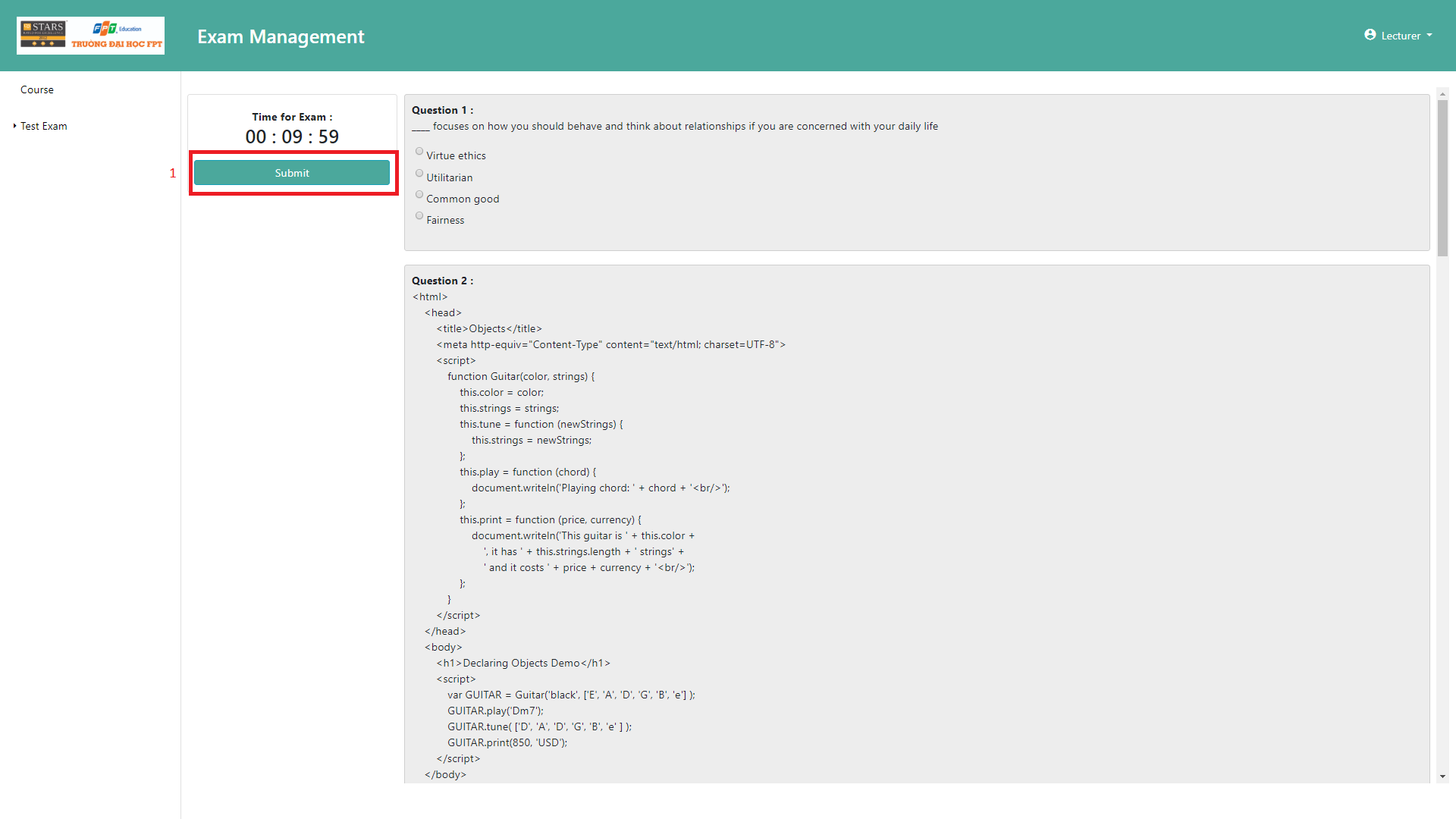


Figure 54 take an exam

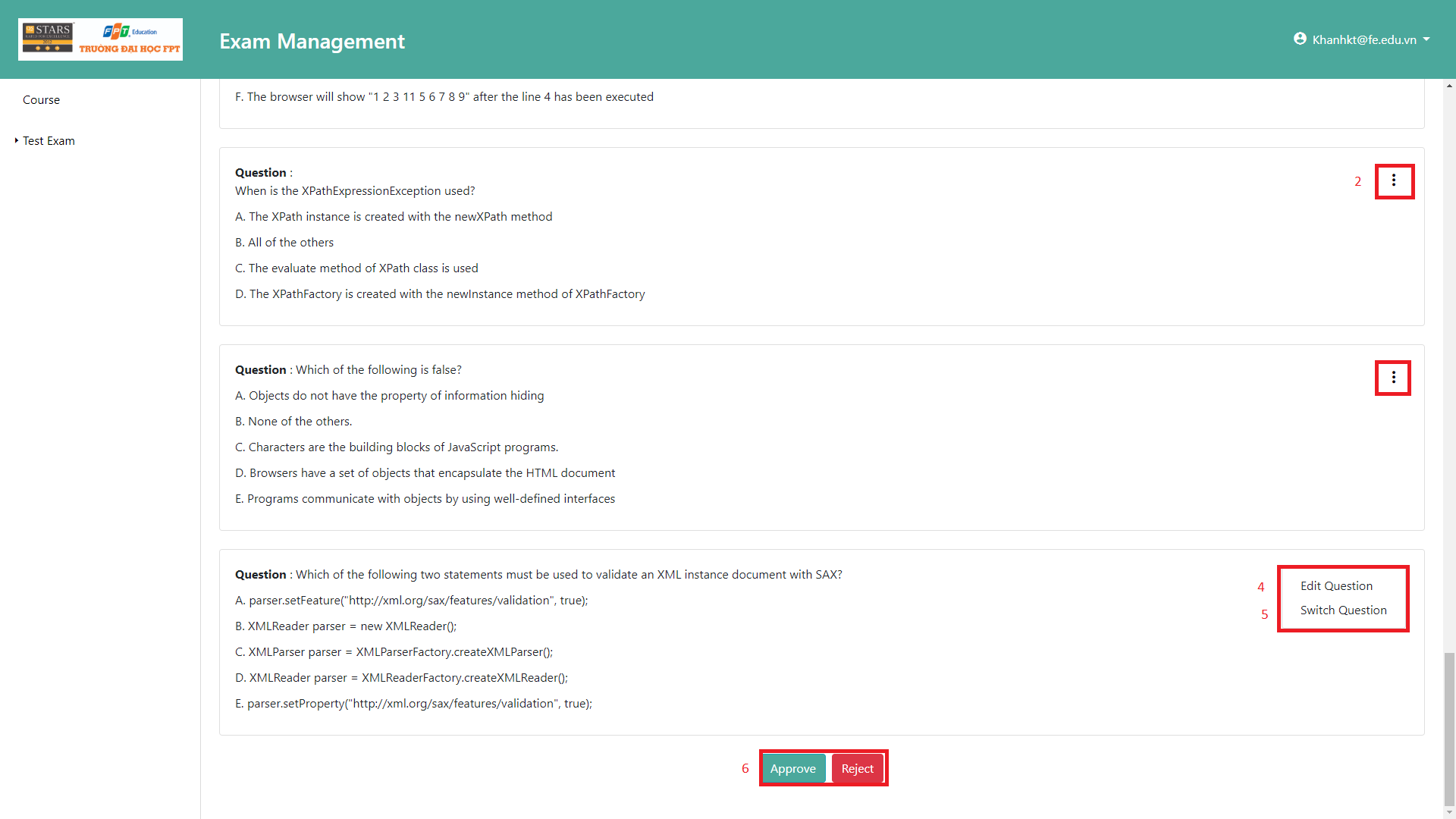


Figure 55 take an exam

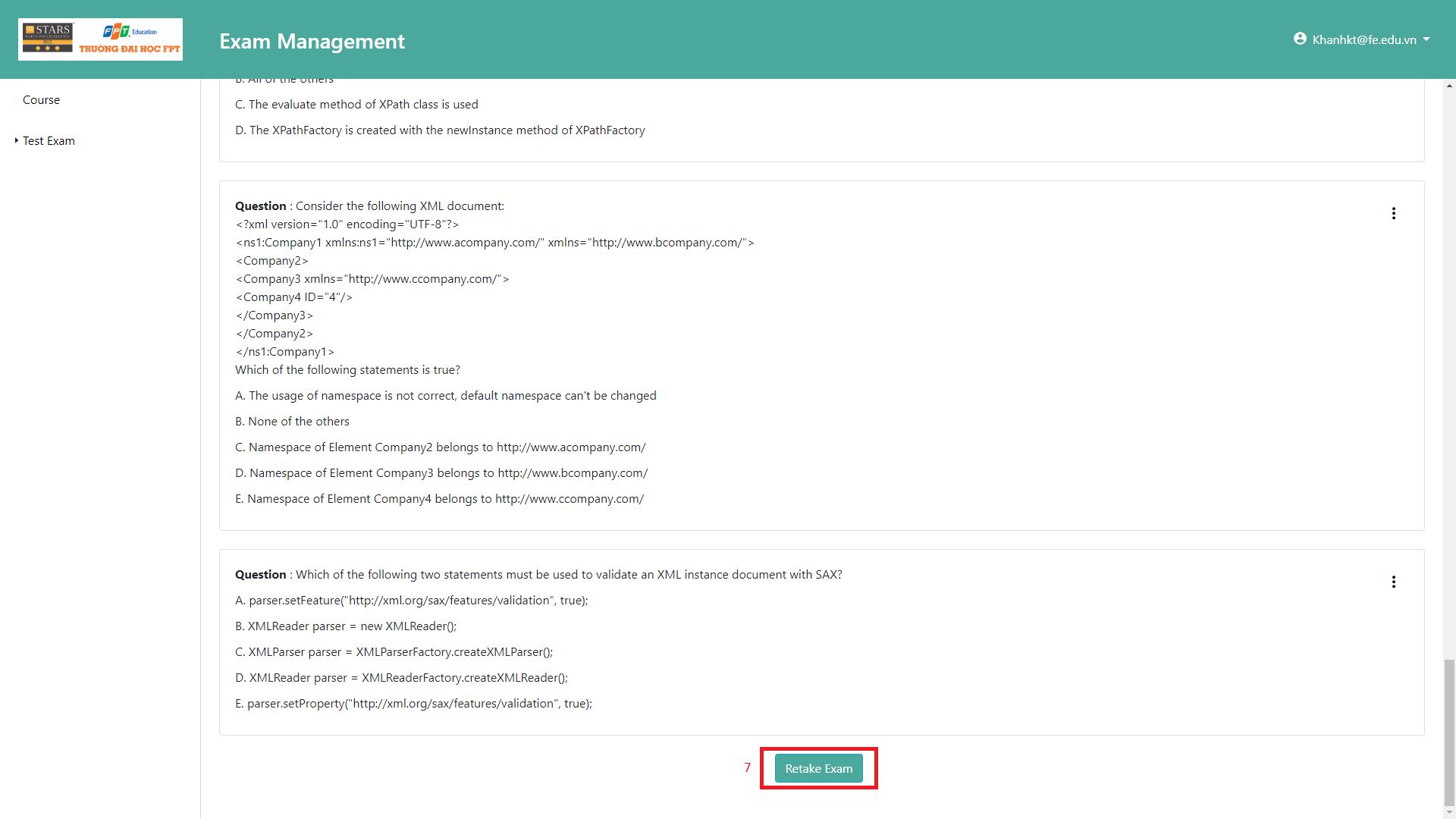


Figure 56 take an exam



Figure 57 take an exam

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 8 | SearchText | Fill in question code or question content to filter question list | No | No | Text | String | 0-255 characters |

Table 88 take an exam fields

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Submit | Send result of exam to server | N/A | Send result to server to grade test exam |
| 2 | ExpandMenu | Open menu which contain edit question and swap question | N/A | Show a small menu. |
| 4 | ShowEditModal | Show a popup which contain current question | N/A | Show a pop up |
| 5 | ShowSwapModal | Show a popup which contain all questions of current course | N/A | Show a pop up |
| 6 | ChangeStatus | Change status test exam to approve/reject | N/A | Status of test exam change to approve/ reject |
| 7 | Retake | Take a test exam again | N/A | Change status test exam to pending and take a test exam again |
| 9 | ShowInformation | Show information of question | N/A | An information of question is expanded below the question row in the table. |
| 10 | SwapQuestion | Swap question in question bank with current question | N/A | Test exam content has been changed and status exam change to edited, button retake exam is shown and hide approve/reject button |

Table 89 Take an exam buttons

### Create Exam

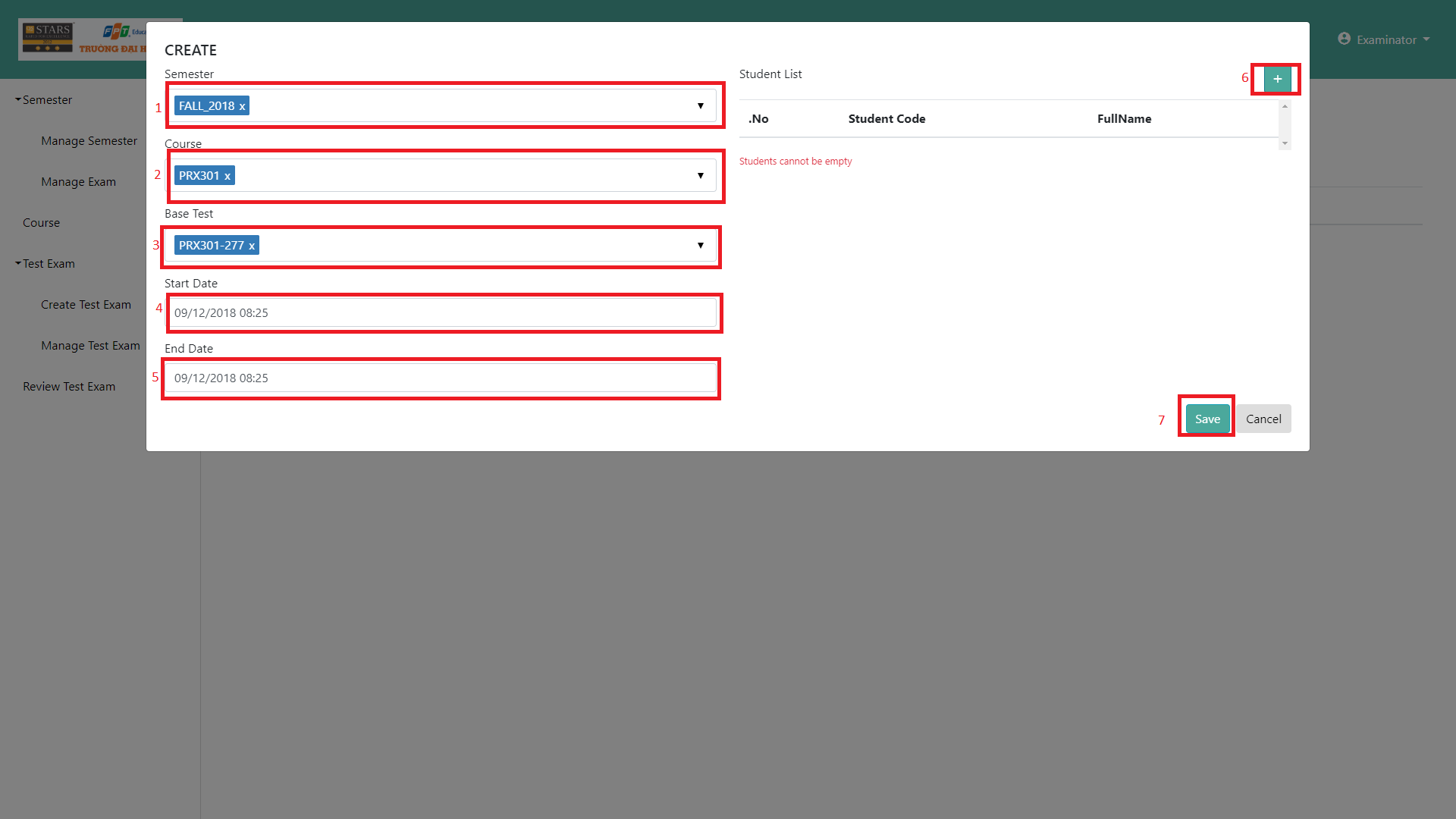


Figure 58 create exam

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | SemesterCode | Fill semester code | No | Yes | Text | String | 0-50 characters |
| 2 | CourdeCode | Fill course code | No | Yes | Text | String | 0-50 characters |
| 3 | BaseTest | Fill basetest | No | Yes | Text | String | 0-20 characters |

Table 90 create exam fields

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 4 | StartDate | Select a date | Pick a date which equal or greater than current date | A date is set |
| 5 | EndDate | Select a date | Pick a date which equal or greater than start date | A date is set |
| 6 | ImportFile | Select a student file. | File must be excel format and have 2 column Fullname and Membercode | A student list will be add to form |
| 7 | Submit | Validate all field and create an exam. | Validate all fields | Send request to create an exam |

Table 91 create exam buttons

### Set Approver

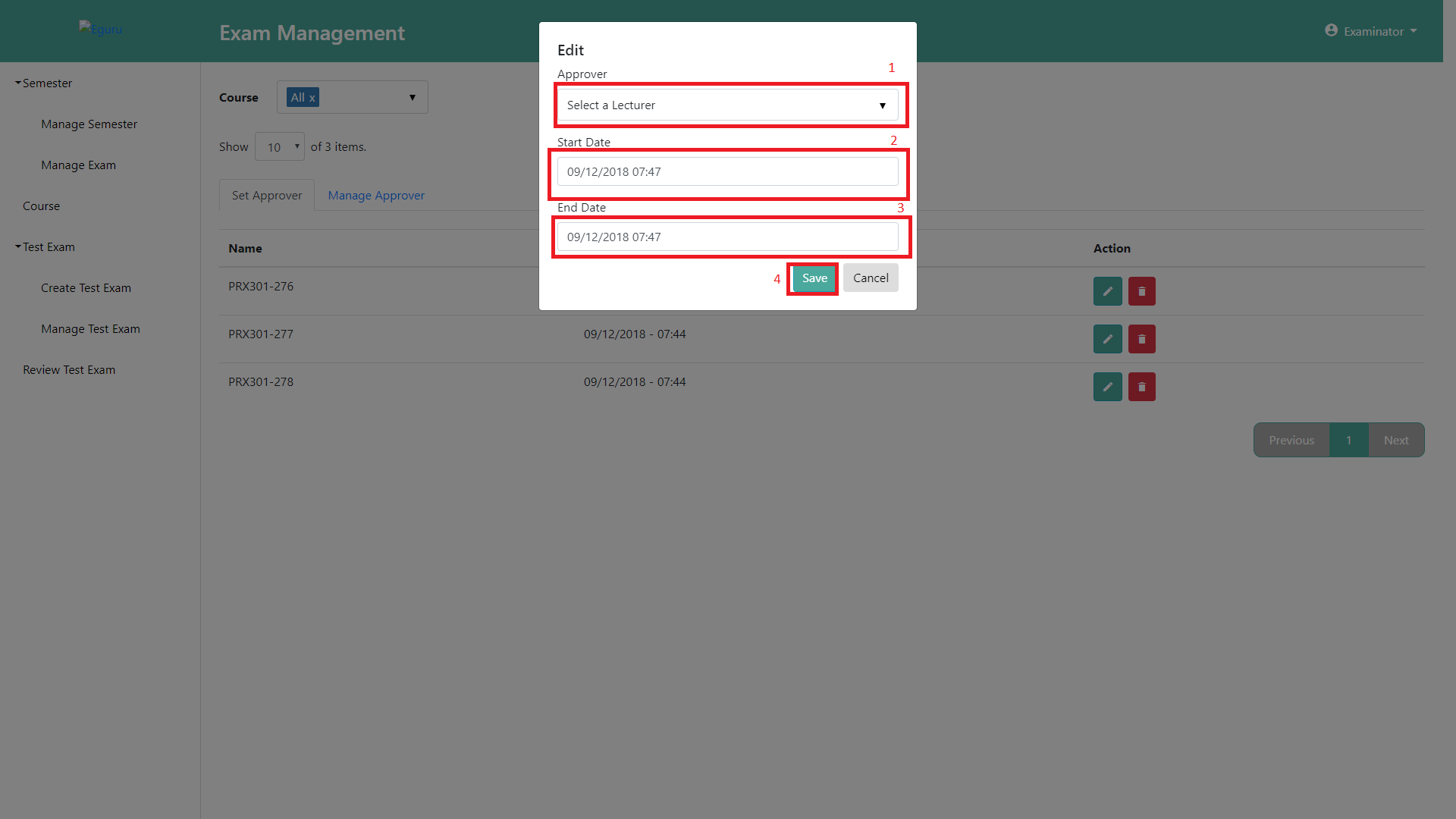


Figure 59 set approver

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Approver | Fill approver | No | Yes | Text | String | 0-50 characters |

Table 92 set approver fields

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | StartDate | Select a date | Pick a date which equal or greater than current date | A date is set |
| 3 | EndDate | Select a date | Pick a date which equal or greater than start date | A date is set |
| 4 | Submit | Validate all field and create an exam. | Validate all fields | Send request to create an exam |

Table 93 set approver buttons

## Database Design

### Entity Relationship Diagram

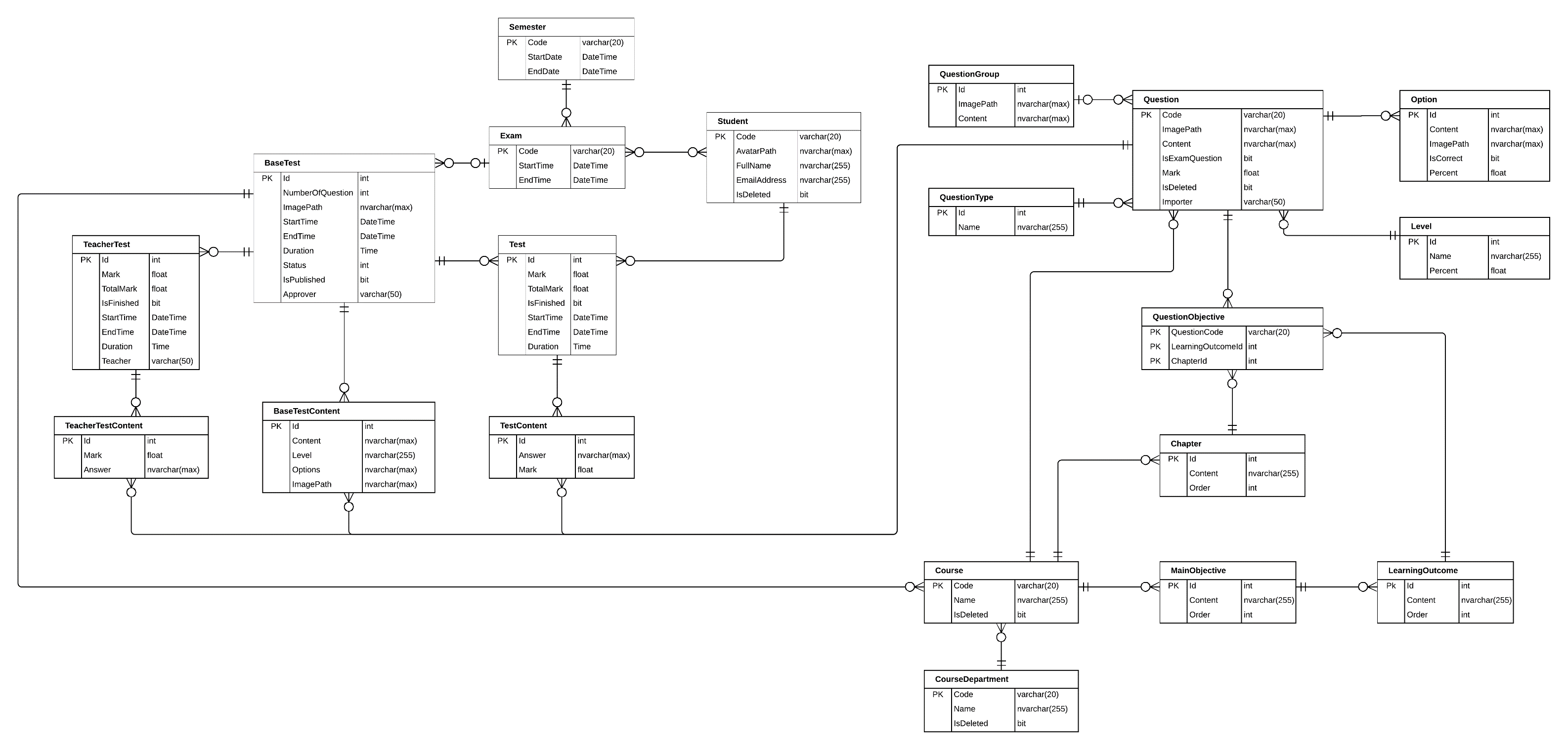


Figure 60 ER DIAGRAM

### 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 94 ERD Dictionary

### ER 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 95 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 96 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 97 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 98 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 99 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 100 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 101 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 102 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 |
| Percent | float | Percent will Minus if choice wrong |

Table 103 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 104 Level table Description

#### QuestionType

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

Table 105 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 106 QuestionGroup table Description

#### QuestionObjective

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

Table 107 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 108 Chapter table Description

#### Course

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

Table 109 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 110 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 111 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 112 CourseDepartment 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 113 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 114 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 algorithm.

Levenshtein distance algorithm is used to identify the differences between the ranges 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

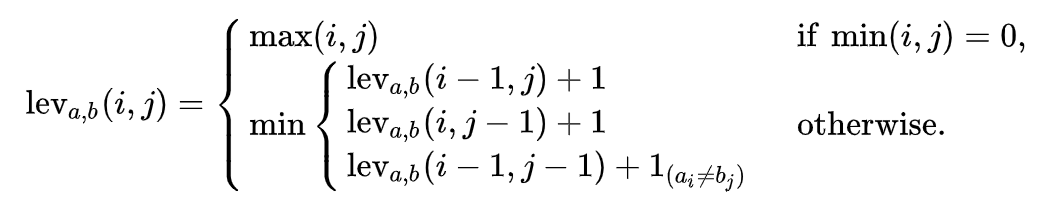
- Replace with another character

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 pseudo code 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):

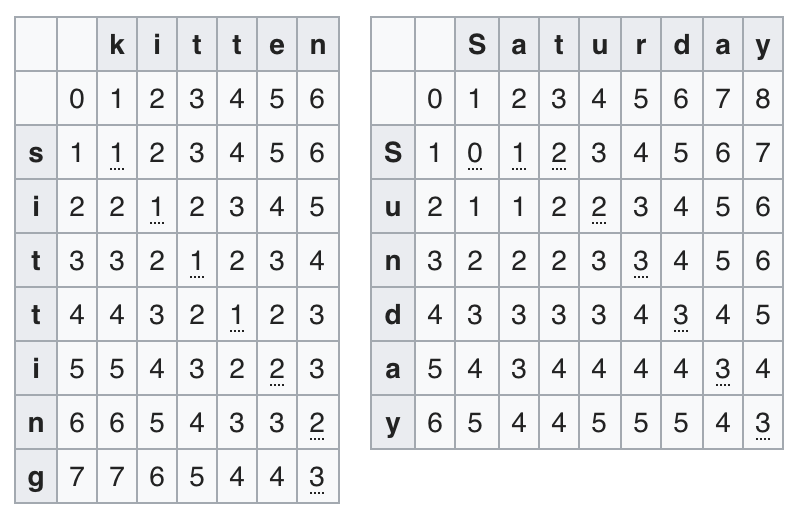


Figure 61 Levenshtein distance example result

#### 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:

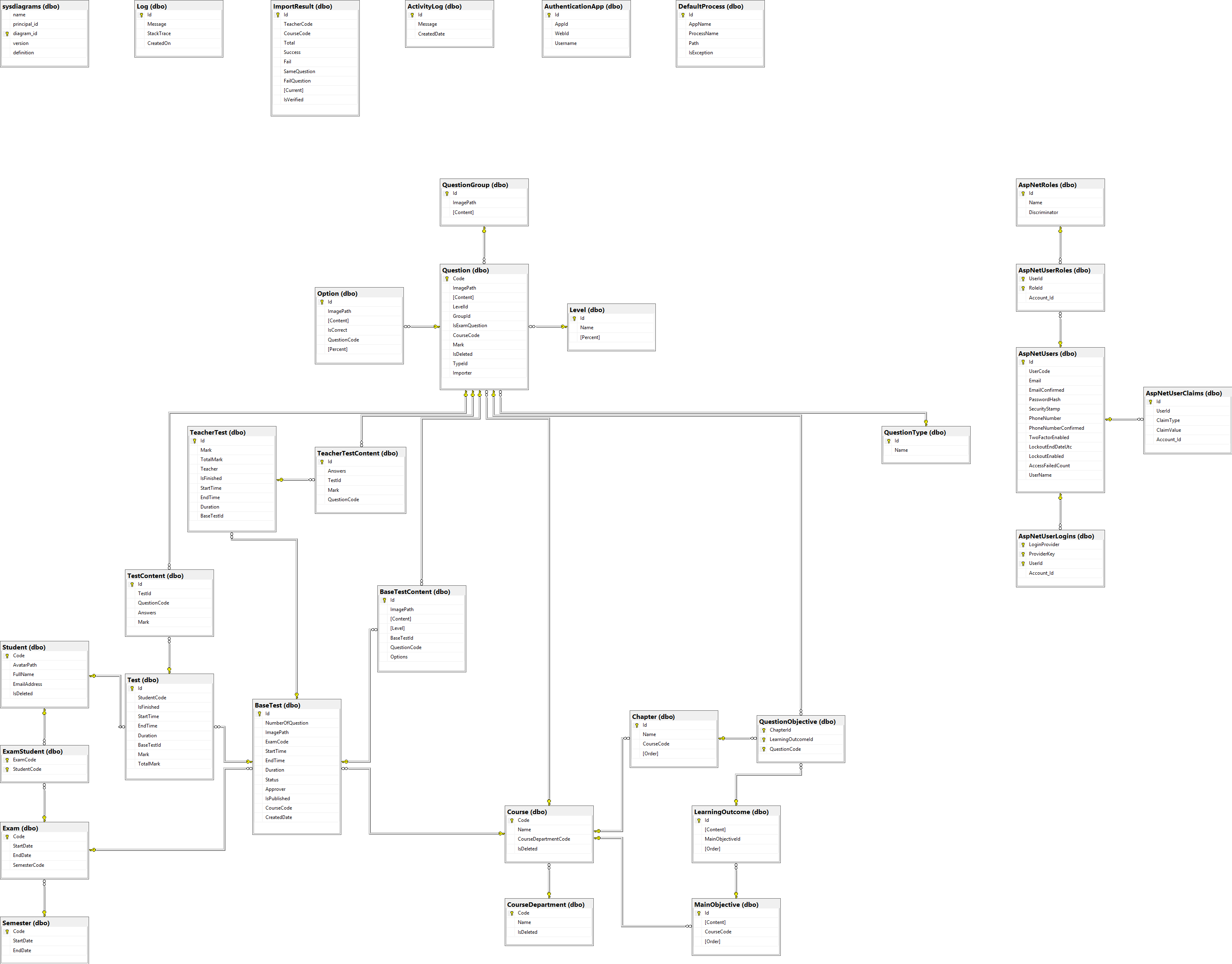
****

Figure 62 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 115 Physical Diagram Description

### Data Dictionary:

#### 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 116 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 |
| SemesterCode | varchar(20) | Code of Semester (ex: Fall 2018) |

Table 117 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 118 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 |
| ExamCode | varchar(20) | Code of Semester (ex: Fall 2018) |
| CourseCode | Int | Course Code |

Table 119 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 |
| baseTestId | int | Base Test Id |
| questionCode | nvarchar(20) | Question of this test exam |

Table 120 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 |
| baseTestId | int | Base Test Id |
| StudentCode | varchar(20) | Code of a Student (ex SE61801) |

Table 121 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 |
| QuestionCode | nvarchar(20) | Question Code |
| TestId | int | Id Student’s Test Exam |

Table 122 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 |
| LevelId | Int | Level of this question |
| TypeId | Int | Type of this question (single choice, multiple choice) |
| GroupId | int | Group question in reading question |

Table 123 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 |
| QuestionCode | nvarchar(20) | Question of this option |

Table 124 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 125 Level table Description

#### QuestionType

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

Table 126 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 127 QuestionGroup table Description

#### QuestionObjective

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

Table 128 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…) |
| CourseCode | Int | Course Code |

Table 129 Chapter table Description

#### Course

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| Code | Int | Course Code |
| Name | nvarchar(255) | Course Name |
| IsDeleted | bit | Is this course deleted |
| CourseDepartmentCode | nvarchar(20) | Department of this course |

Table 130 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 131 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 |
| MainObjectiveId | Int | MainObjective Id |

Table 132 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 133 CourseDepartment 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 |
| baseTestId | int | Base test of this test exam |

Table 134 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 |
| questionCode | nvarchar(20) | Question of this answer |

Table 135 TeacherTestContent 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 136 AuthenticationApp table Description

#### Import Result

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Id | Import Result Id |
| TeacherCode | Teacher who import that section |
| CourseCode | Course import |
| Total | Number of question |
| Success | Number of imported success |
| Fail | Number of imported fail |
| SameQuestion | Number of duplicate question |
| FailQuestion | Number of parse fail question |

Table 137 ImportResult table 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 138 DefaultProcess table 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

- Prevent other applications

- Remember Student’s answer

- Prevent multiple logged

### 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

## 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 139 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 140 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 141 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 142 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 143 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 144 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 145 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 146 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 147 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 148 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 149 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 150 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.

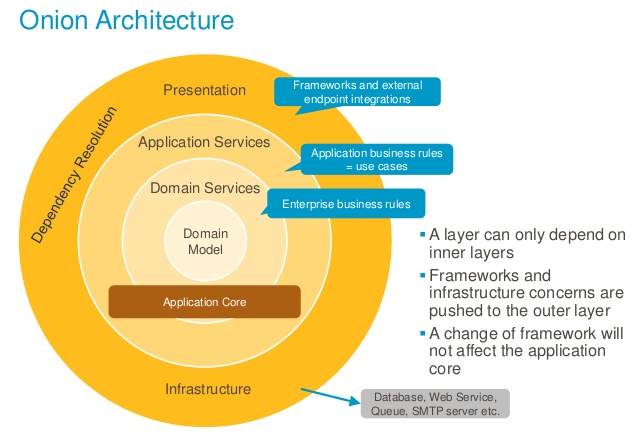


Figure 63 Onion Architecture

### 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.

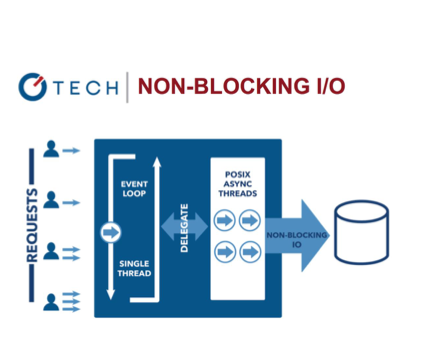


Figure 64 Event-driven architecture

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

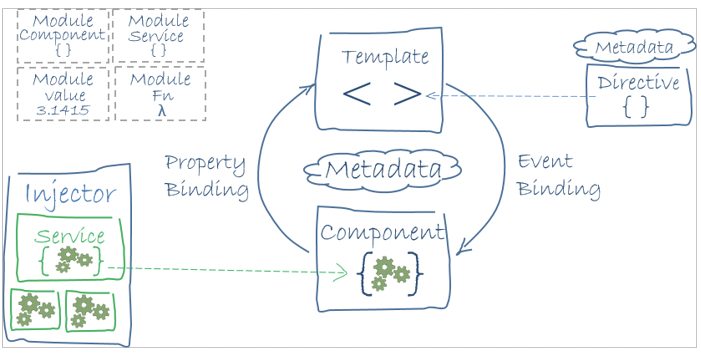


Figure 65 Angular Component Example

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

# Software User’s Manual

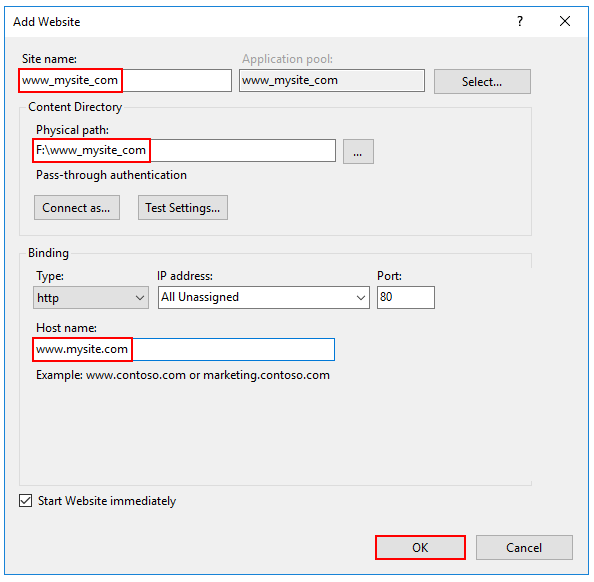
## Installing Guide

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.

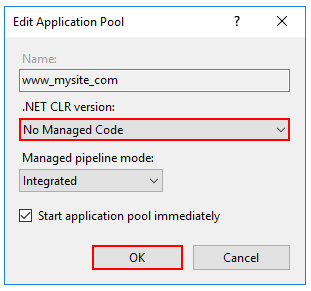
**Important**

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.

Reference: “https://docs.microsoft.com/en-us/aspnet/web-forms/overview/deployment/visual-studio-web-deployment/deploying-to-iis”

## User Guide

### Admin Create Semester

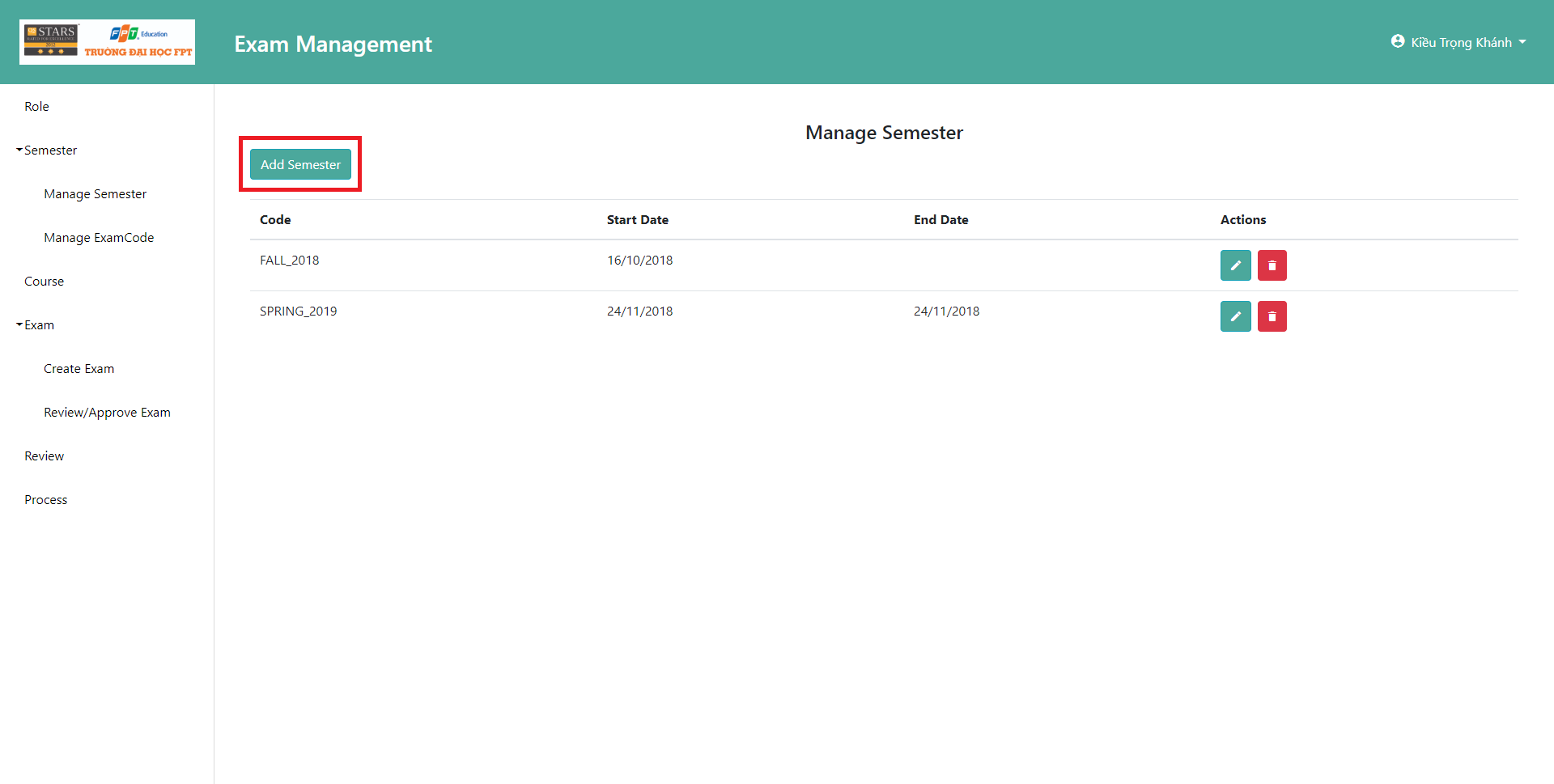


Figure 66 Guide create semester 1

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

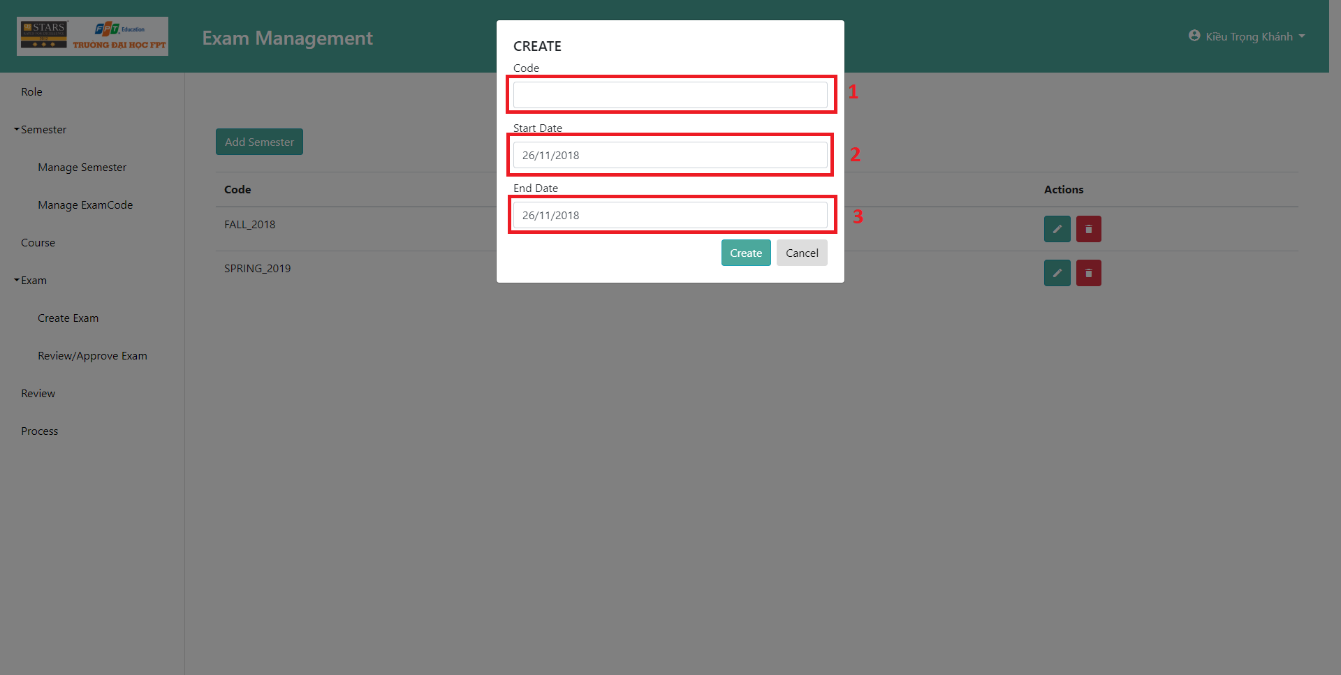


Figure 67 Guide create semester 2

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

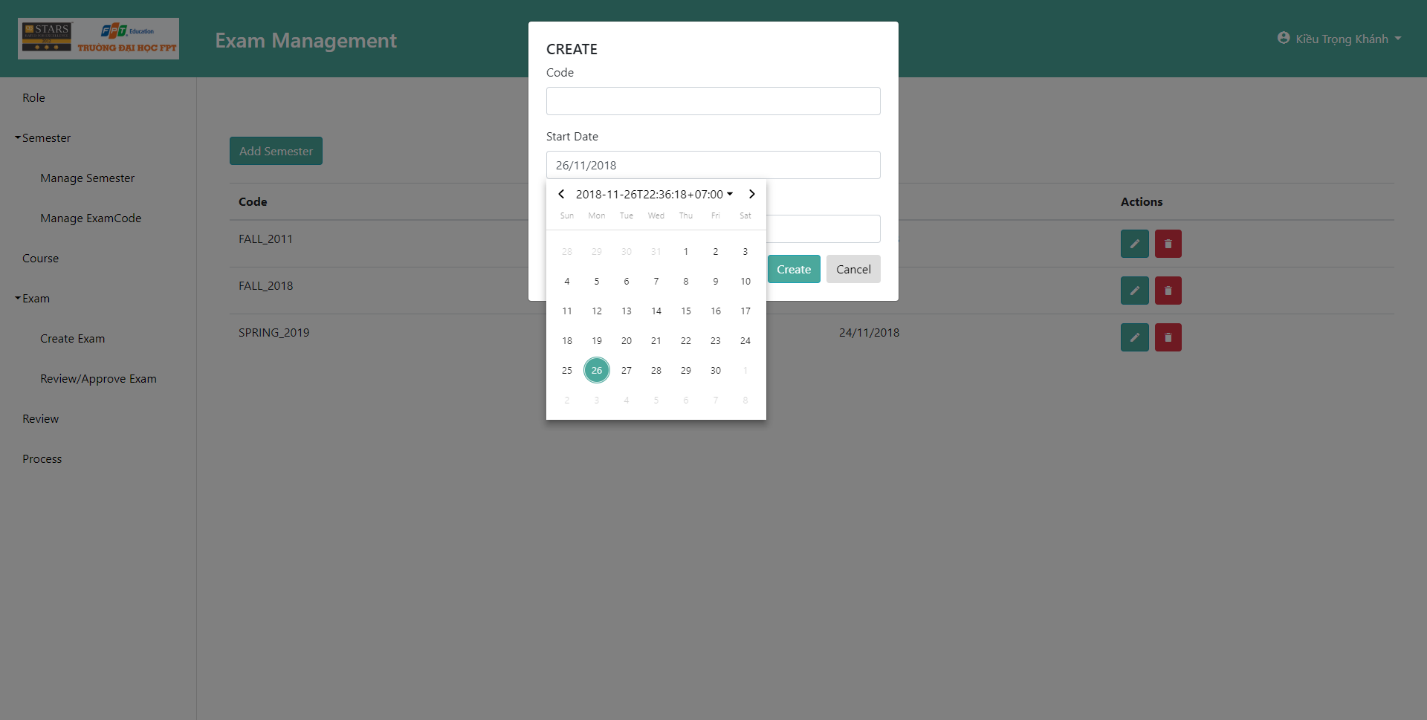


Figure 68 Guide create semester 3

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

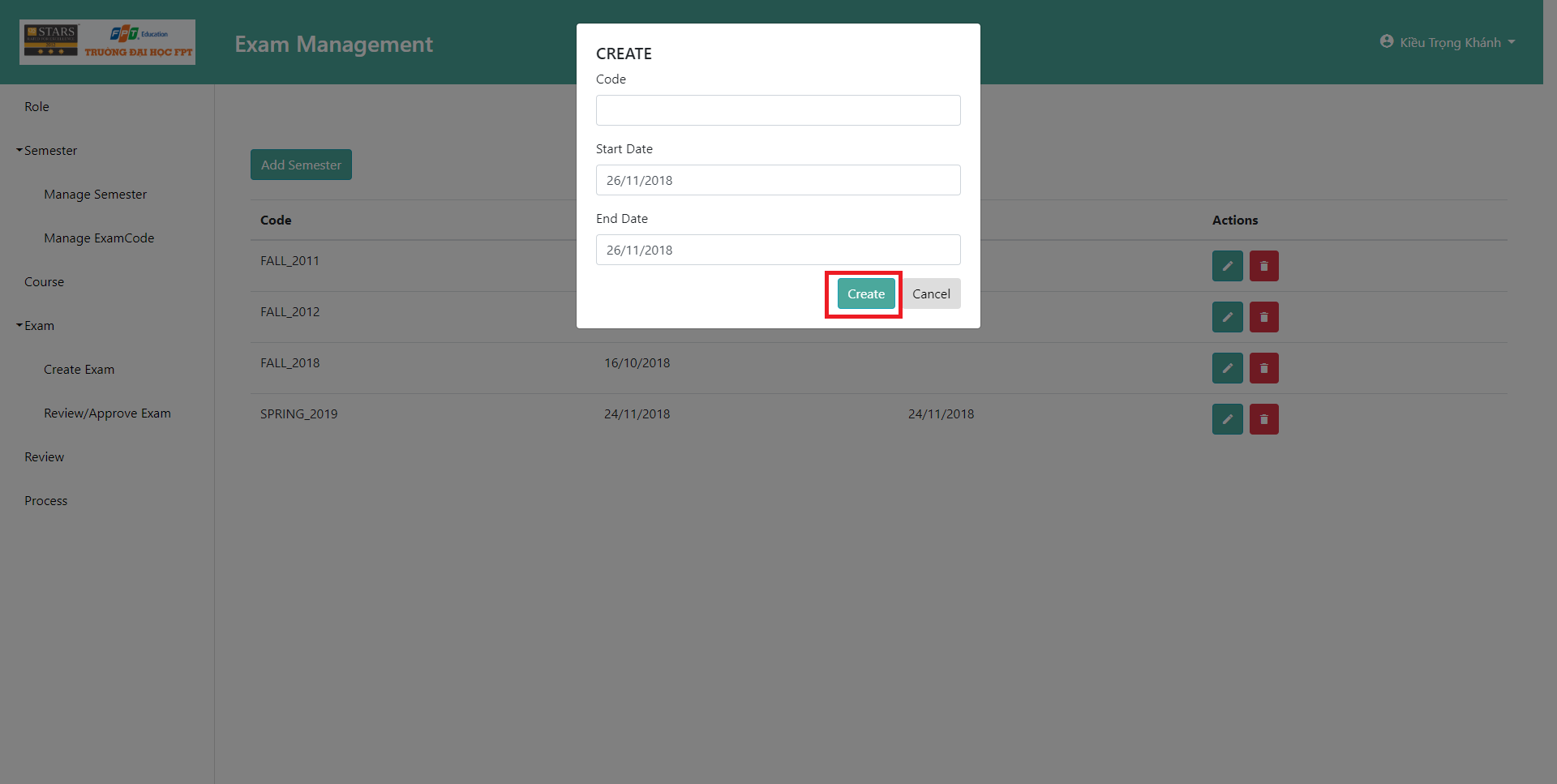


Figure 69 Guide create semester 4

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

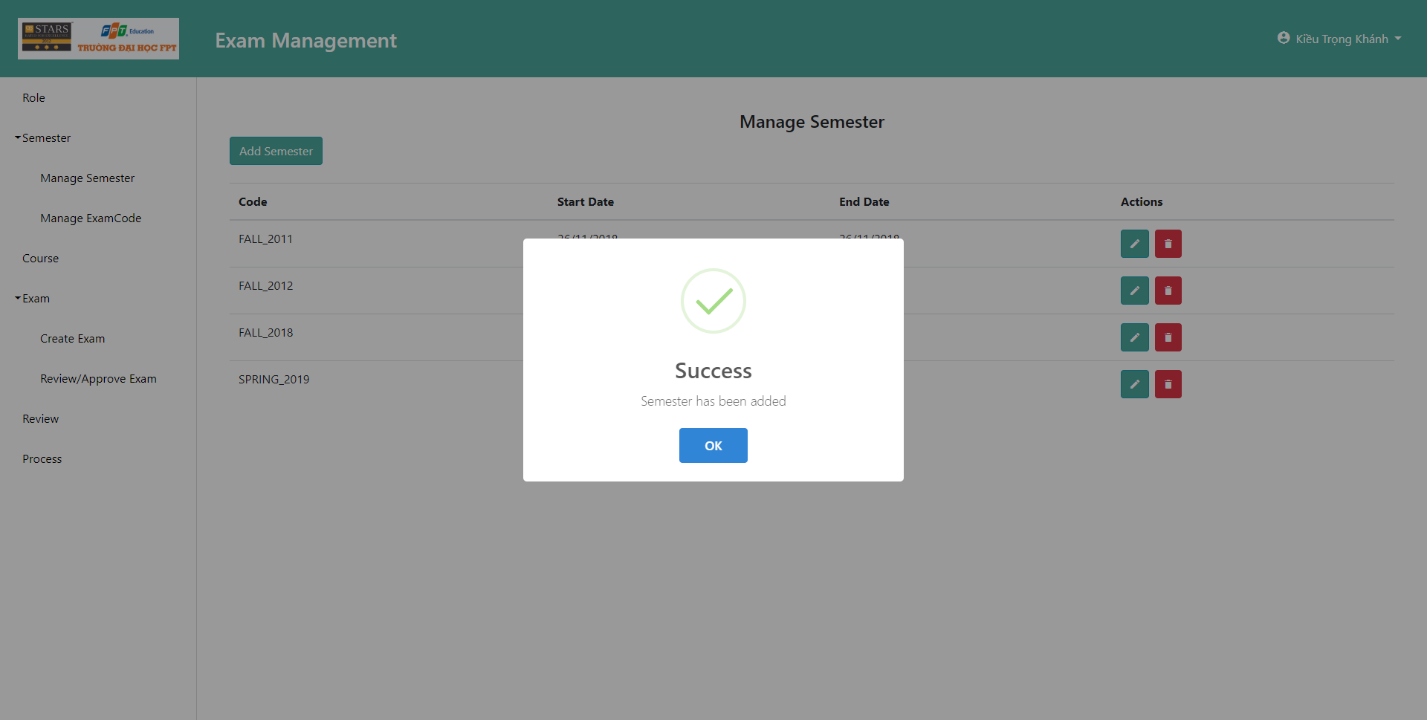


Figure 70 Guide create semester 5

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

### Teacher approve/reject exam

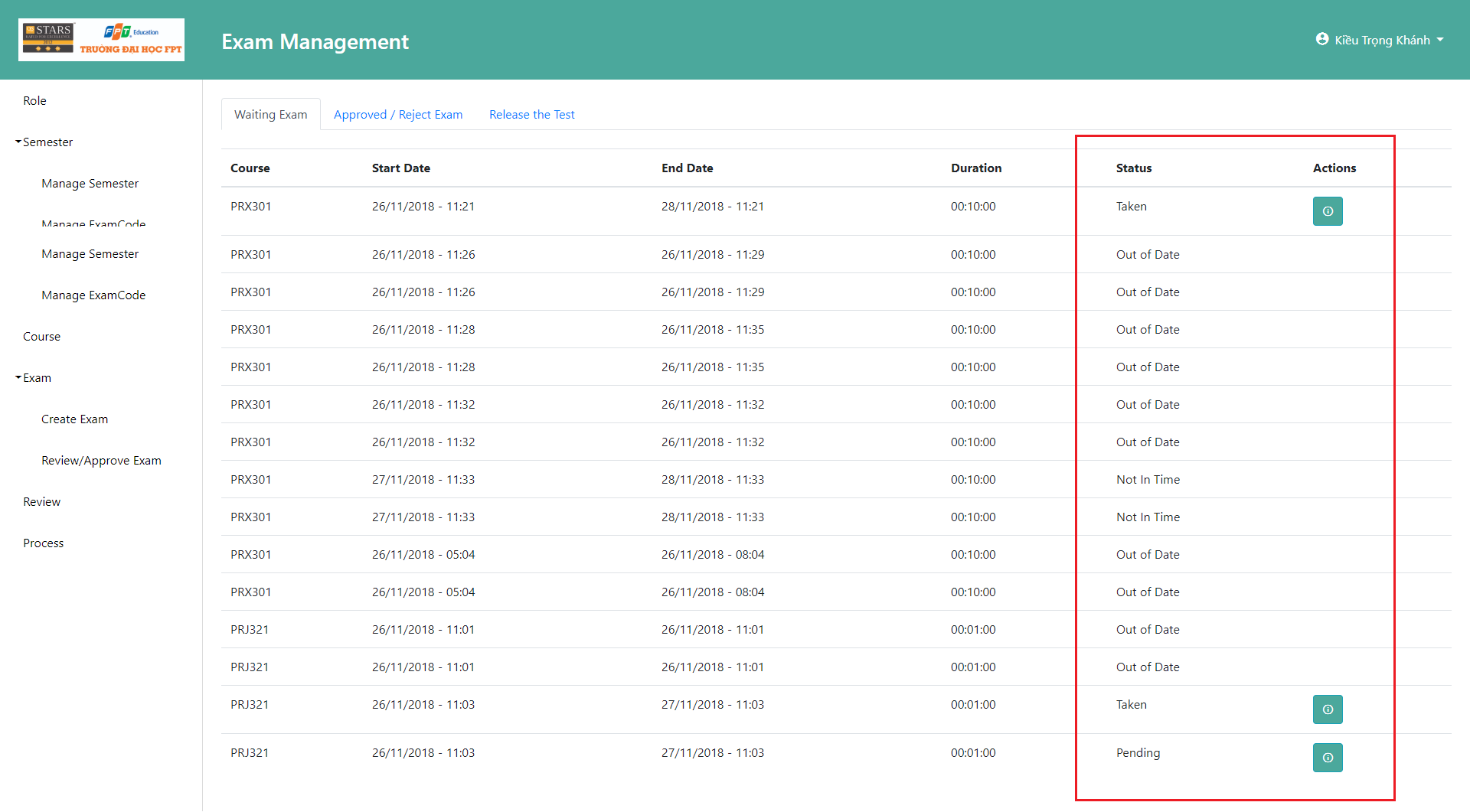


Figure 71 Guide teacher approve test exam

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.

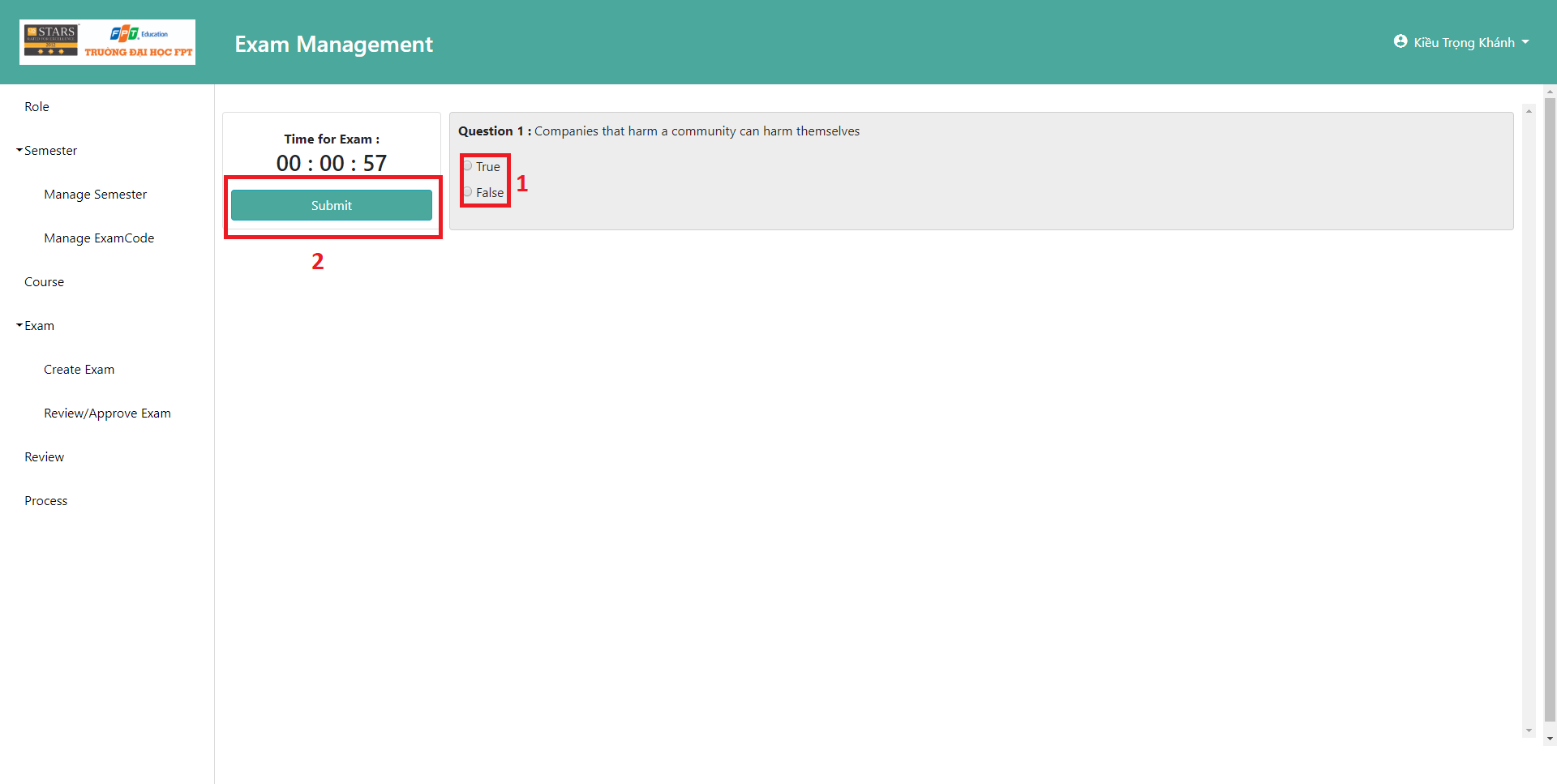


Figure 72 Guide teacher approve test exam 2

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

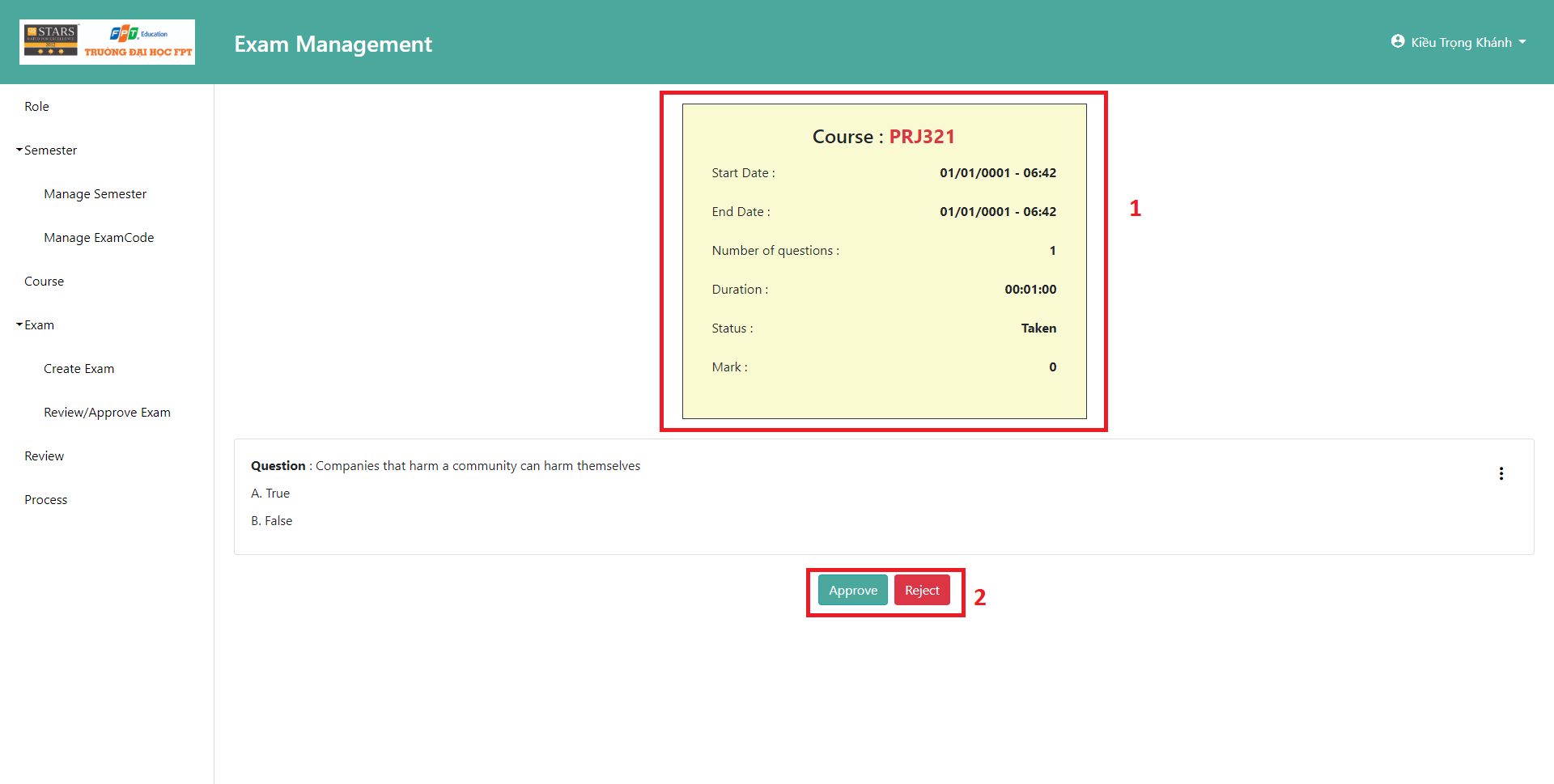


Figure 73 Guide teacher approve test exam 3

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

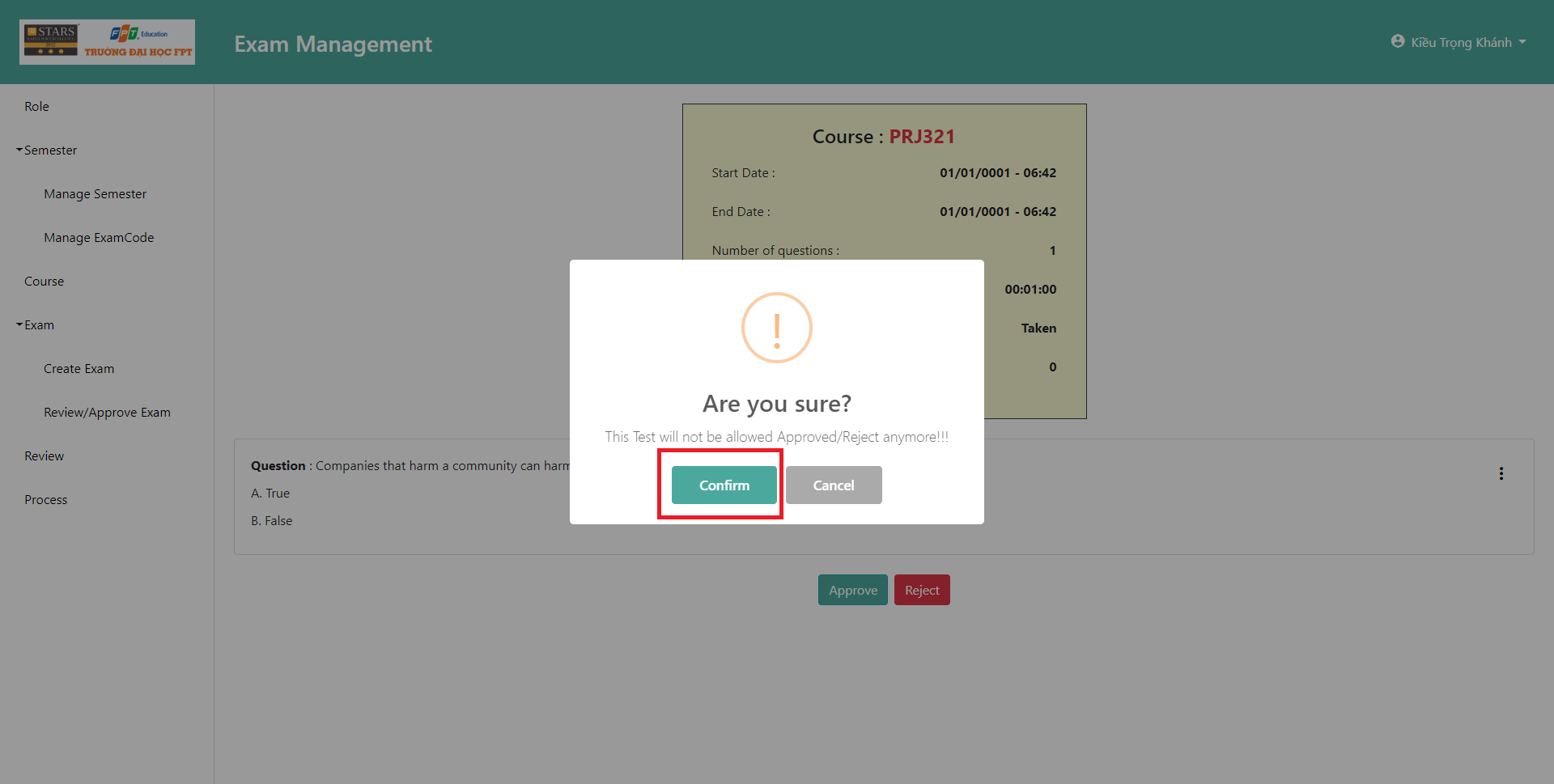


Figure 74 Guide teacher approve test exam 4

Click confirm to Approve or Reject the Test Exam.

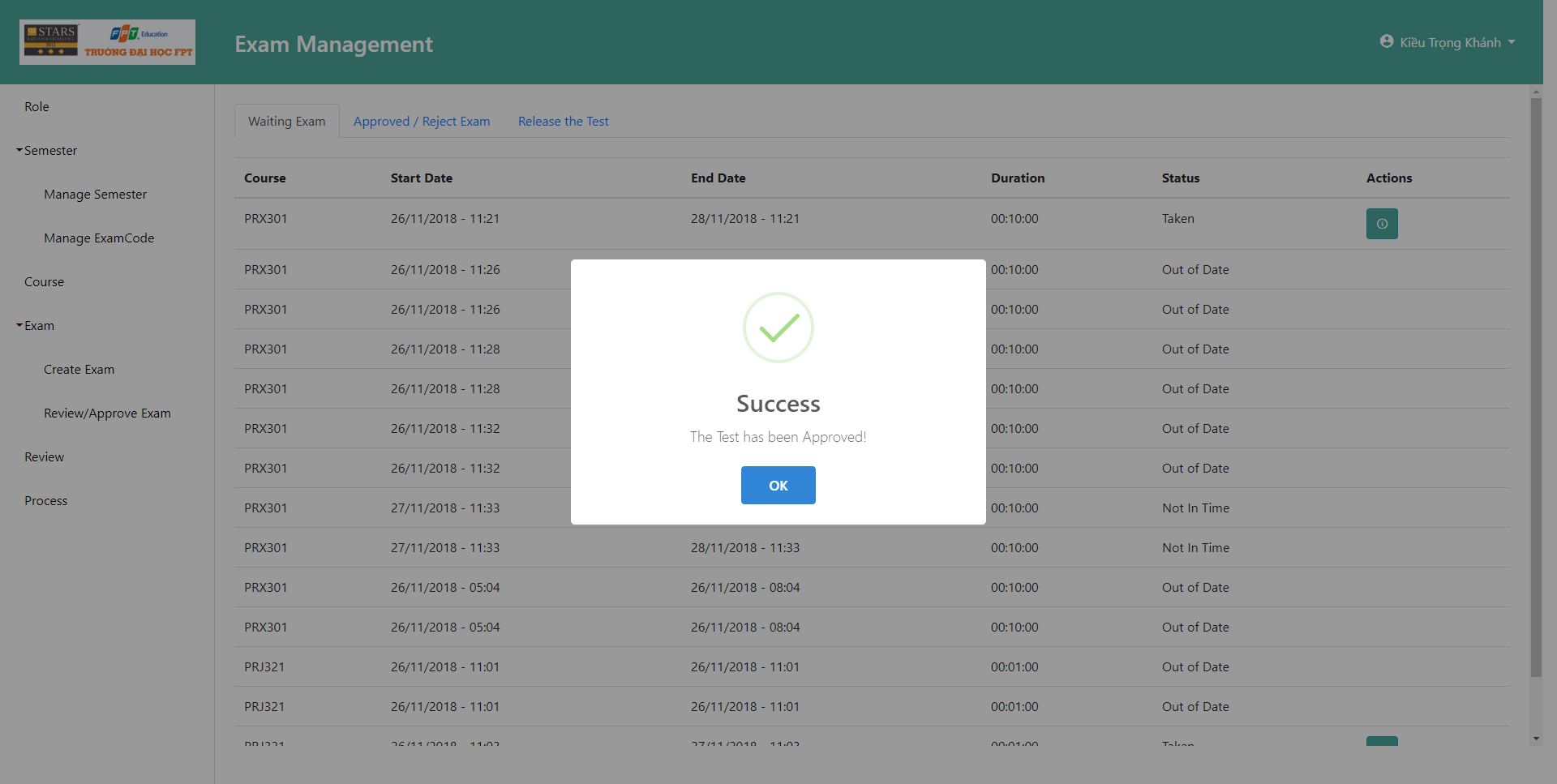


Figure 75 Guide teacher approve test exam 5

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

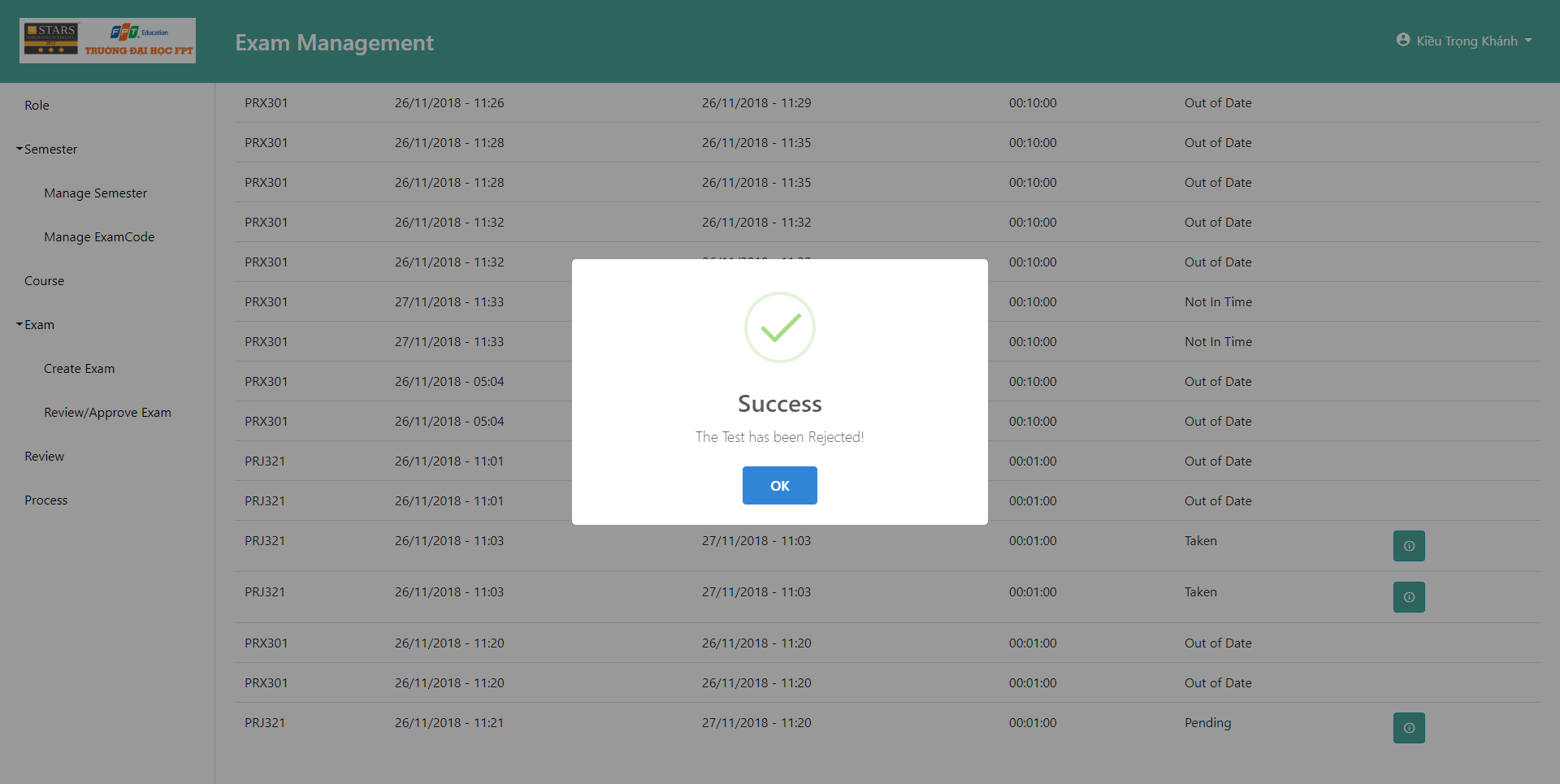


Figure 76 Guide teacher approve test exam 6

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

### Staff Create Exam

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



Figure 77 Guide Staff create exam 1

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.

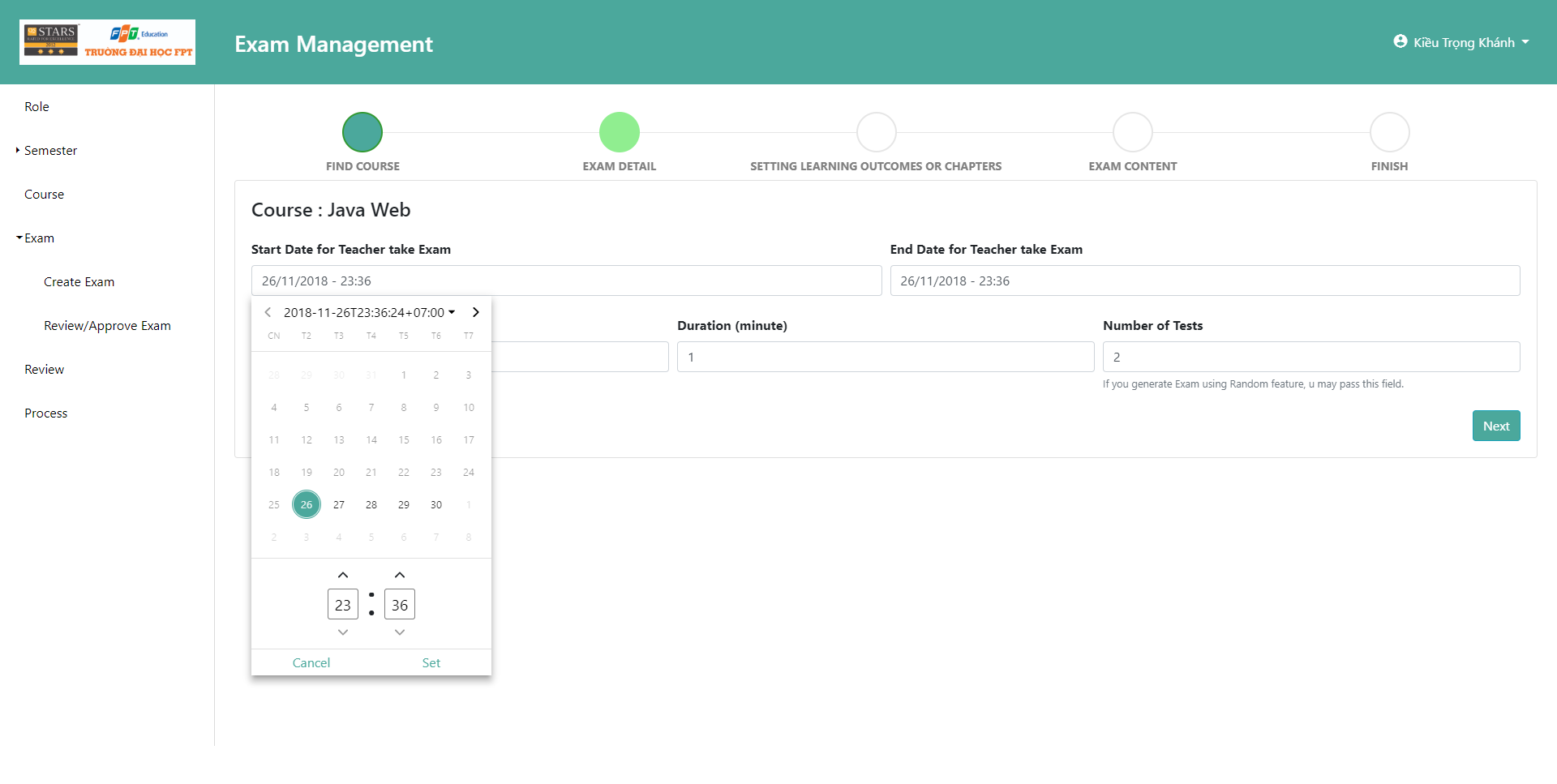


Figure 78 GUIDE STAFF CREATE EXAM 2

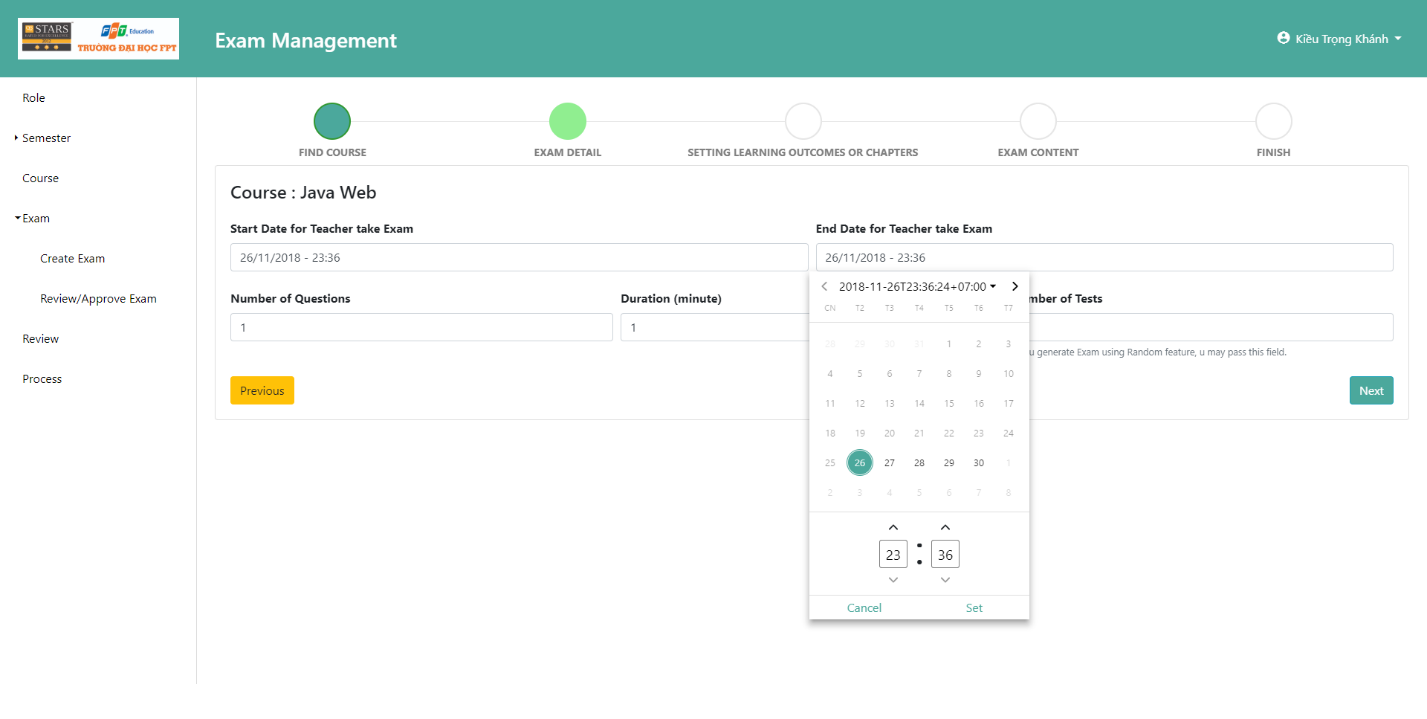


Figure 79 GUIDE STAFF CREATE EXAM 3

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

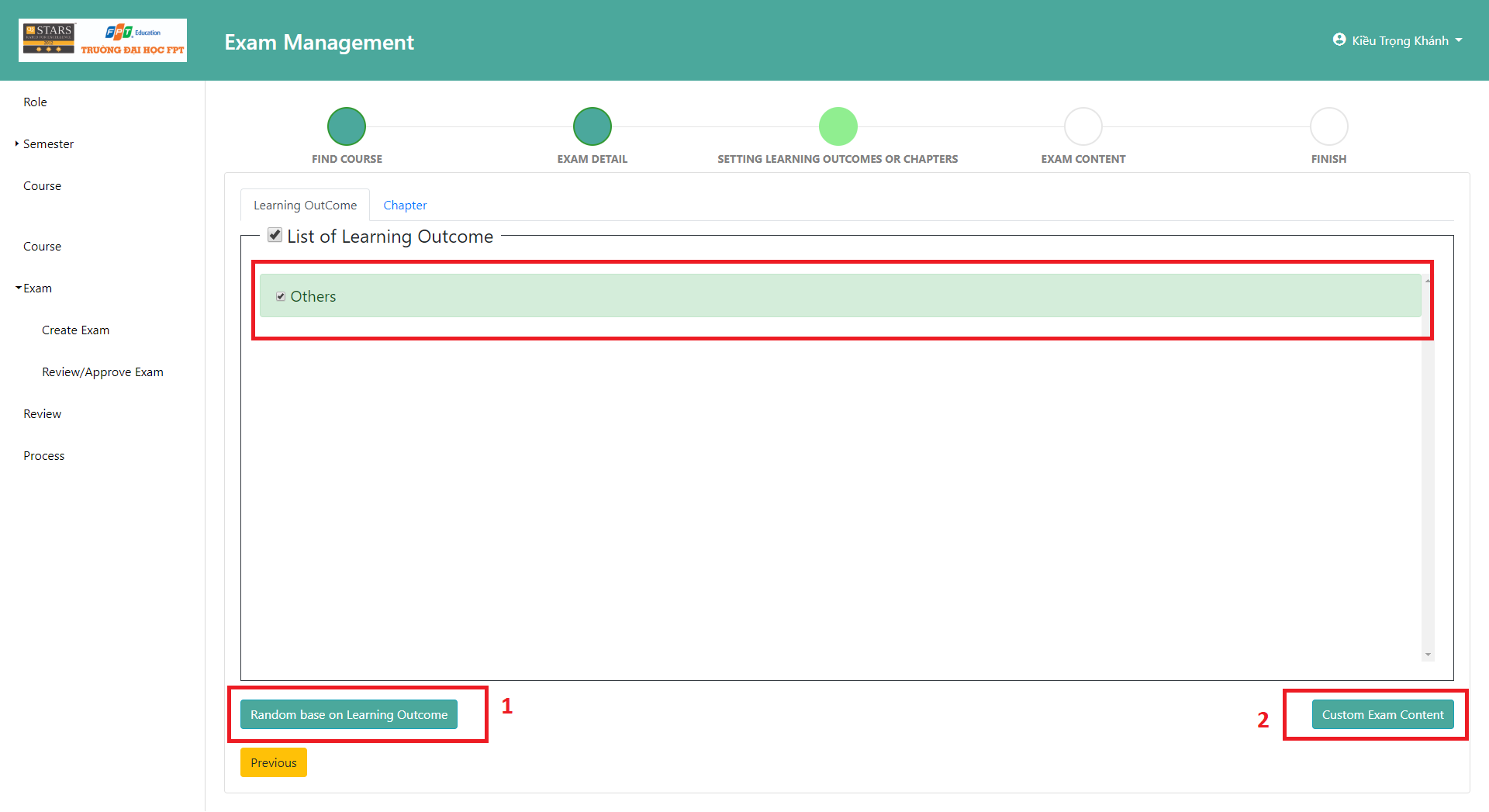


Figure 80 GUIDE STAFF CREATE EXAM 4

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).

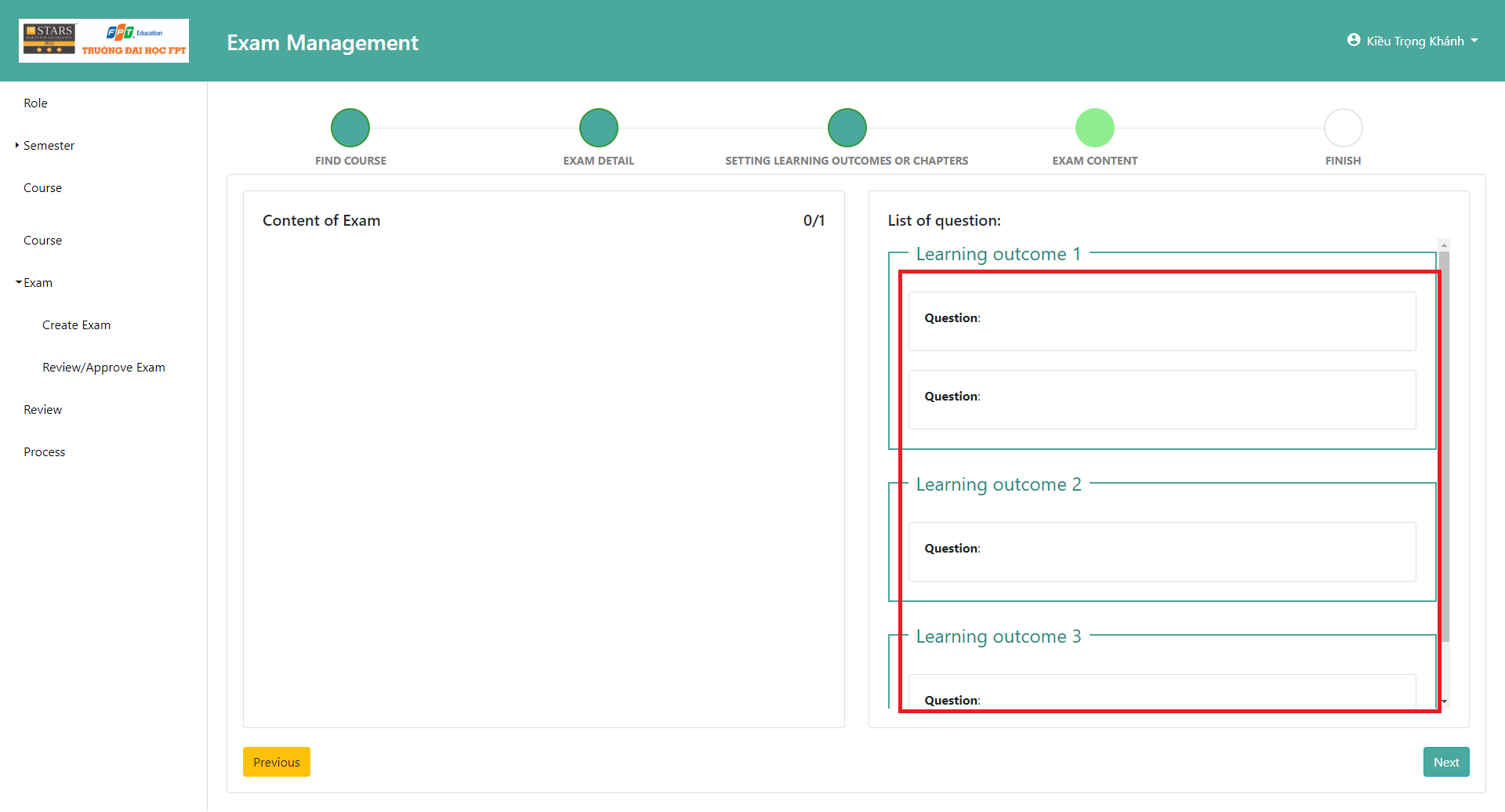


Figure 81 GUIDE STAFF CREATE EXAM 5

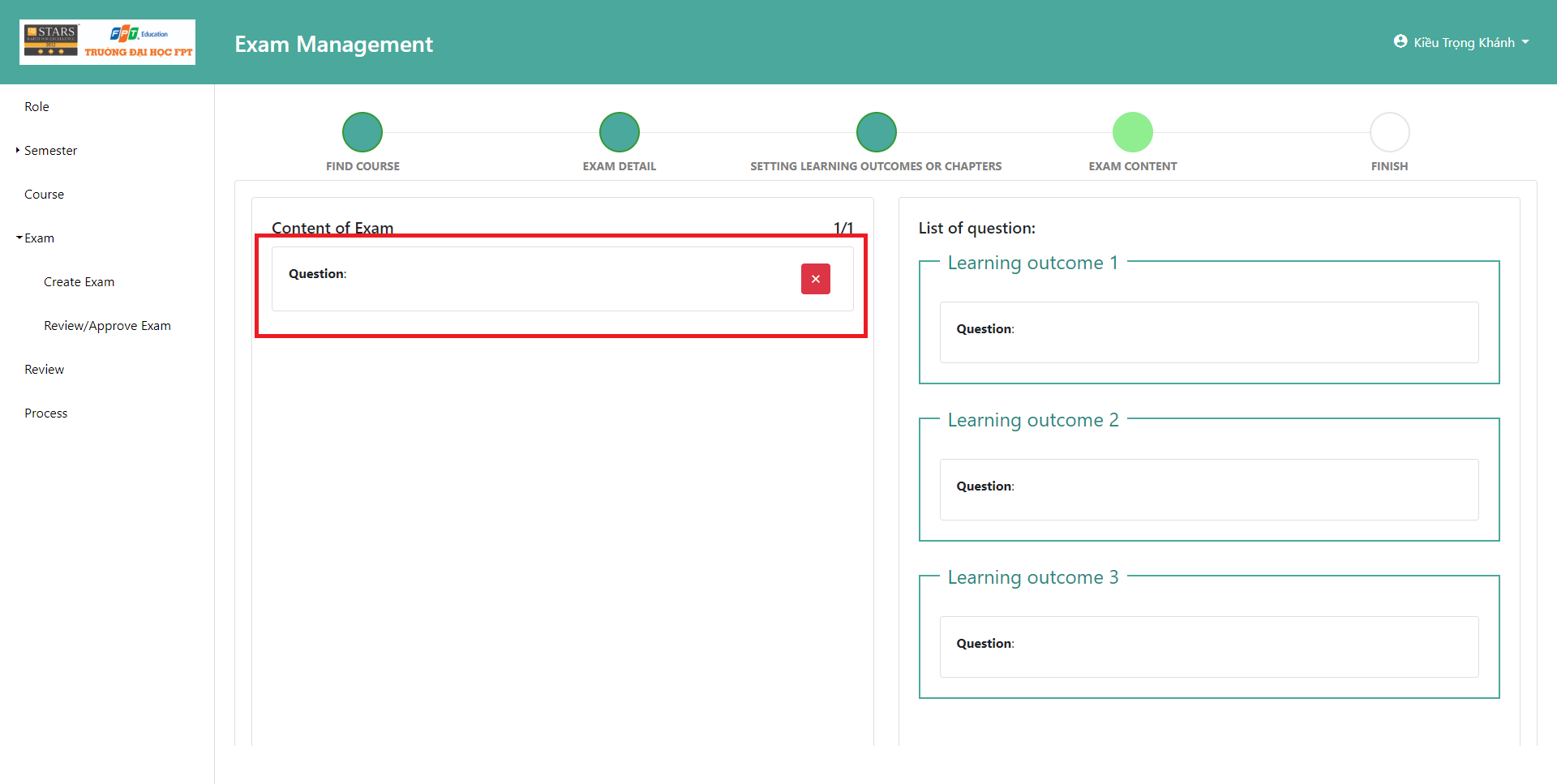


Figure 82 GUIDE STAFF CREATE EXAM 6

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

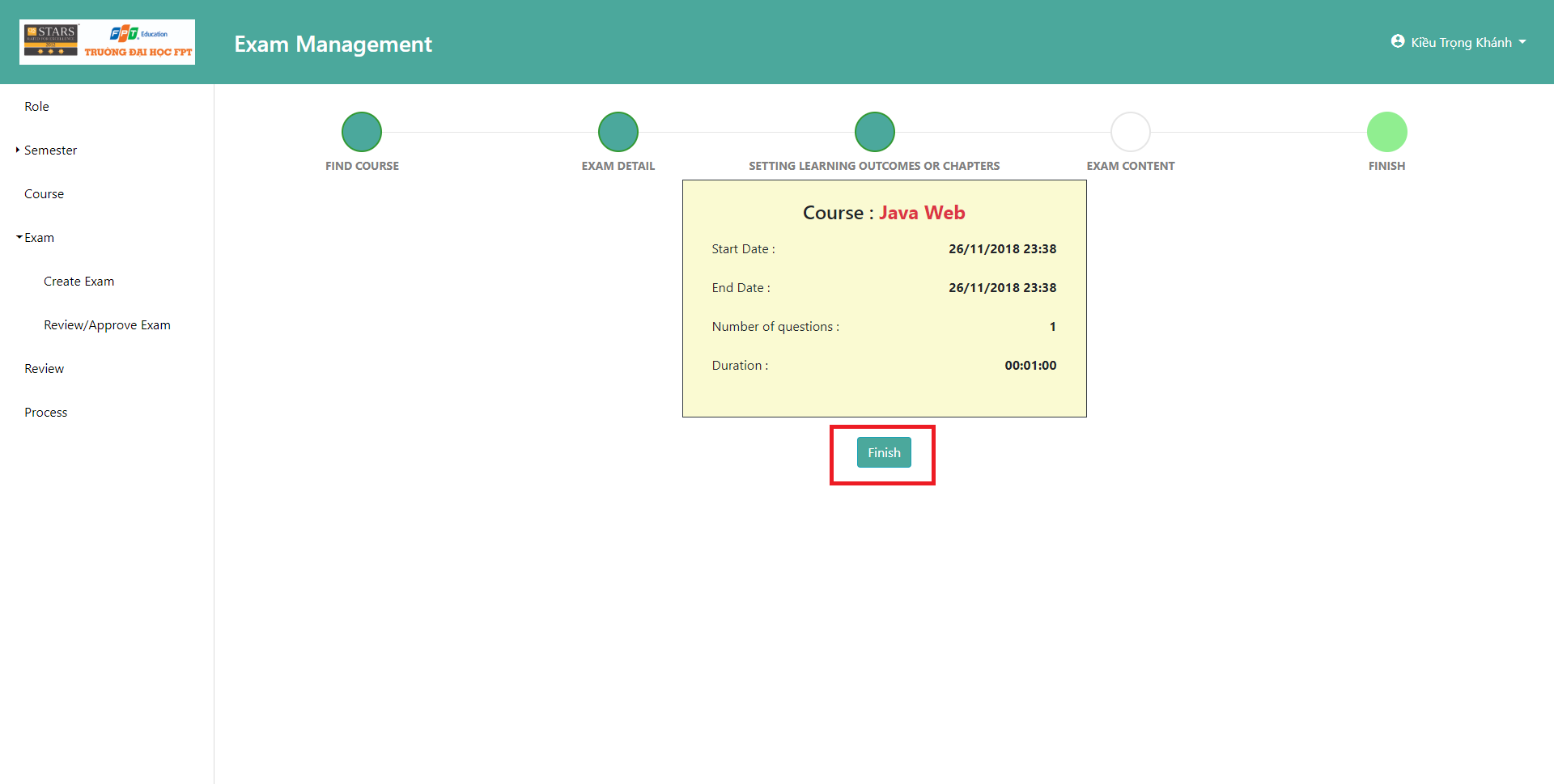


Figure 83 GUIDE STAFF CREATE EXAM 7

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