DriveIT

Korbela-Pokryvka-Stefkovic

Table of Contents

Table of Figures

Stereotypes

UseCase		
Tagged Value Definitions	**Level**	
	Type	Enumeration
	Enumeration Values	Summary, User, Subfunction
	Complexity	
	Type	Enumeration
	Enumeration Values	Low, Medium, High
	Use Case Status	
	Type	Enumeration
	Enumeration Values	Name Only, Initial, Base, Complete, Deferred
	Implementation Status	
	Type	Enumeration
	Enumeration Values	Scheduled, Started, Partially Complete, Complete, Partially Deferred
	Preconditions	
	Type	Multi-line Text
	Post-conditions	
	Type	Multi-line Text
	Author	
	Type	Text
	Assumptions	
	Type	Multi-line Text

Case Story

requirement		
Tagged Value Definitions	**Text**	
	Type	HTML
	ID	
	Type	Text
	source	
	Type	Text
	kind	
	Type	Enumeration
	Enumeration Values	Functional, Performance, Interface
	verifyMethod	
	Type	Enumeration
	Enumeration Values	Analysis, Demonstration, Inspection, Test
	risk	
	Type	Enumeration
	Enumeration Values	High, Medium, Low
	status	
	Type	Enumeration
	Enumeration Values	Proposed, Approved, Rejected, Deferred, Implemented, Mandatory, Obsolete

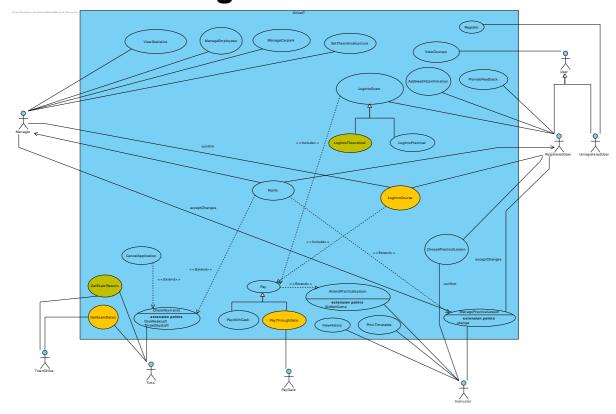**Include**

Extend

use

ORM Abstract Persistable

control

entity

boundary

Use Case Diagram

Summary

Name	Description
👤 Manager	
👤 TownOffice	
👤 Time	
👤 PayGate	
👤 Instructor	
👤 RegisteredUser	
👤 User	
👤 UnregisteredUser	
⬬ GetExamResults	
⬬ ViewStatistics	
⬬ ManageEmployees	
⬬ ManageCarpark	
⬬ ViewHistory	
⬬ PrintTimetable	
⬬ ManagePracticalLesson	
⬬ SetTheoreticalLecture	
⬬ GetExamDates	Main Flow of Events:

	1. The use case starts when current time reached one week from last event. 2. System checks if exam dates have been published by Town Office. 3. IF exam dates are accessible 3.1 System downloads exam dates to its database. 3.2 System sets exam dates in database as available. Alternative Flows: If exam dates were not accessible, Manager is informed and Time starts counting another one week period.
🔵 LogIntoPractical	
🔵 LogIntoTheoretical	
🔵 LogIntoExam	
🔵 ChoosePracticalLesson	
🔵 AttendPracticalLesson	
🔵 ProvideFeedback	
🔵 AddHealthConfirmation	
🔵 CancelApplication	
🔵 Notify	
🔵 CheckPayments	
🔵 PayThroughGate	Main Flow of Events: 1. The use case starts when Pay is invoked by another use case and the gate option is chosen. 2. RegisteredUser is redirected to the PayGate. 3. RegisteredUser is asked to provide his/her credentials. 4. IF the credentials are recognized and correct 4.1 RegisteredUser is presented with a form of payment to fill in. 4.2 IF RegisteredUser chooses "Pay", 4.2.1 PayGate tries to proceed with the payment. 4.2.2 IF the payment was successful 4.2.2.1 RegisteredUser is notified about successful operation and money is transferred. 4.2.3 ELSE 4.2.3.1 RegisteredUser is notified about a problem. 4.3 ELSE IF RegisteredUser chooses "Cancel" 4.3.1 Operation is canceled. Alternative flows: If the credentials were not recognized or incorrect, RegisteredUser is prompted again.
🔵 PayWithCash	
🔵 Pay	
🔵 LogIntoCourse	Main Flow of Events: 1. The use case starts when a RegisteredUser chooses "Log Into Course". 2. System returns lists of available courses. 3. RegisteredUser chooses a course. 4. IF RegisteredUser meets course preconditions 4.1. System displays price and other details. 4.2. INCLUDE Pay

	4.3. IF Payment was successful 4.3.1. Send application to Manager for approval. 4.3.2. IF Manager approves 4.3.2.1. Assign RegisteredUser to course. Alternate Flows: If RegisteredUser does not meet course preconditions, system display message and returns to previous menu. If Payment was not successful three days before course starts, application is canceled. If Manager does not approve, application is canceled.
Register	
ViewCourses	
DriveIT	

Details

♀ Manager

♀ TownOffice

♀ Time

♀ PayGate

♀ Instructor

♀ RegisteredUser

♀ User

♀ UnregisteredUser

⊣ N/A

⊣ N/A

⊣ N/A

GetExamResults

ViewStatistics

ManageEmployees

ManageCarpark

ViewHistory

PrintTimetable

ManagePracticalLesson

Extension Points

change

SetTheoreticalLecture

GetExamDates

Details

Name	Value
Preconditions	One week period has passed. Exam dates have been published by Town Office.
Post-conditions	Exam dates are inserted into system.

LogIntoPractical

LogIntoTheoretical

LogIntoExam

ChoosePracticalLesson

AttendPracticalLesson

Extension Points

DidNotCome

ProvideFeedback

N/A

AddHealthConfirmation

CancelApplication

● Notify

● CheckPayments
Extension Points

OneWeekLeft

ThreeDaysLeft

● PayThroughGate
Details

Name	Value
Preconditions	RegisteredUser is client of online banking.
Post-conditions	none

● PayWithCash

● Pay

● LogIntoCourse
Details

Name	Value
Preconditions	RegisteredUser is logged in the system.
Post-conditions	RegisteredUser is enrolled in a course.

● Register

● ViewCourses

▯ DriveIT
Children

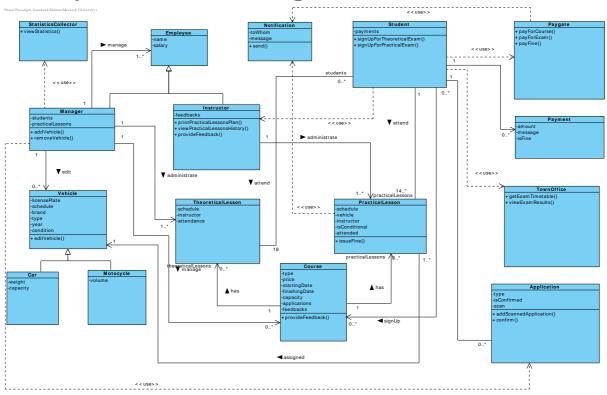| Name	Description
● ViewCourses	
● Register	
● LogIntoCourse	Main Flow of Events: 1. The use case starts when a RegisteredUser chooses "Log Into Course". 2. System returns lists of available courses. 3. RegisteredUser chooses a course. 4. IF RegisteredUser meets course preconditions 4.1. System displays price and other details. 4.2. INCLUDE Pay 4.3. IF Payment was successful

	4.3.1. Send application to Manager for approval. 4.3.2. IF Manager approves 4.3.2.1. Assign RegisteredUser to course. Alternate Flows: If RegisteredUser does not meet course preconditions, system display message and returns to previous menu. If Payment was not successful three days before course starts, application is canceled. If Manager does not approve, application is canceled.
⬤ Pay	
⬤ PayWithCash	
⬤ PayThroughGate	Main Flow of Events: 1. The use case starts when Pay is invoked by another use case and the gate option is chosen. 2. RegisteredUser is redirected to the PayGate. 3. RegisteredUser is asked to provide his/her credentials. 4. IF the credentials are recognized and correct 4.1 RegisteredUser is presented with a form of payment to fill in. 4.2 IF RegisteredUser chooses "Pay", 4.2.1 PayGate tries to proceed with the payment. 4.2.2 IF the payment was successful 4.2.2.1 RegisteredUser is notified about successful operation and money is transferred. 4.2.3 ELSE 4.2.3.1 RegisteredUser is notified about a problem. 4.3 ELSE IF RegisteredUser chooses "Cancel" 4.3.1 Operation is canceled. Alternative flows: If the credentials were not recognized or incorrect, RegisteredUser is prompted again.
⬤ CheckPayments	
⬤ Notify	
⬤ CancelApplication	
⬤ AddHealthConfirmation	
⬤ ProvideFeedback	
⬤ AttendPracticalLesson	
⬤ ChoosePracticalLesson	
⬤ LogIntoExam	
⬤ LogIntoTheoretical	
⬤ LogIntoPractical	
⬤ GetExamDates	Main Flow of Events: 1. The use case starts when current time reached one week from last event. 2. System checks if exam dates have been published by Town Office.

	3. IF exam dates are accessible 3.1 System downloads exam dates to its database. 3.2 System sets exam dates in database as available. Alternative Flows: If exam dates were not accessible, Manager is informed and Time starts counting another one week period.
🔵 SetTheoreticalLecture	
🔵 ManagePracticalLesson	
🔵 PrintTimetable	
🔵 ViewHistory	
🔵 ManageCarpark	
🔵 ManageEmployees	
🔵 ViewStatistics	
🔵 GetExamResults	

Analysis Class Diagram

Visual Paradigm Standard Edition(Masaryk University)

StatisticsCollector
+ viewStatistics()

Employee
-name
-salary

Notification
-toWhom
-message
+ send()

Student
-payments
+ signUpForTheoreticalExam()
+ signUpForPracticalExam()

Paygate
+ payForCourse()
+ payForExam()
+ payFine()

< < use > >

manage
1..*

< < use > >

students
0..*

< < use > >

Manager
-students
-practicalLessons
+ addVehicle()
+ removeVehicle()

Instructor
-feedbacks
+ printPracticalLessonsPlan()
+ viewPracticalLessonsHistory()
+ provideFeedback()

administrate

attend

Payment
-amount
-message
-isFine

0..*

< < use > >

TownOffice
+ getExamTimetable()
+ viewExamResults()

edit

0..*

Vehicle
-licensePlate
-schedule
-brand
-type
-year
-condition
+ editVehicle()

administrate

attend

TheoreticalLesson
-schedule
-instructor
-attendance

PracticalLesson
-schedule
-vehicle
-instructor
-isConditional
-attended
+ issueFine()

1..* practicalLessons
14..* practicalLessons

practicalLessons
0..*
1..*

18

Car
-weight
-capacity

Motocycle
-volume

theoreticalLessons
manage

has

Course
-type
-price
-startingDate
-finishingDate
-capacity
-applications
-feedbacks
+ provideFeedback()

has

signUp

Application
-type
-isConfirmed
-scan
+ addScannedApplication()
+ confirm()

0..*

assigned

< < use > >

Summary

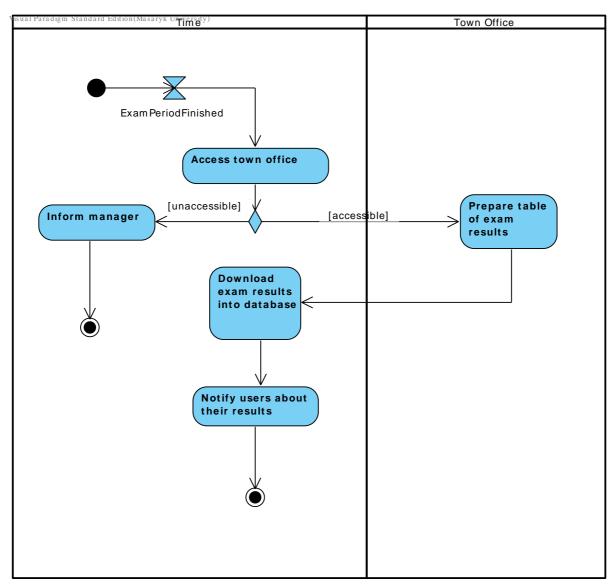| Name	Description
StatisticsCollector	
Student	
Paygate	
Notification	
Employee	
Instructor	
Manager	
Payment	
TownOffice	
Vehicle	
TheoreticalLesson	
PracticalLesson	
Course	
Motocycle	
Car	
Application	

Description

DriveIT Analysis Class Diagram

Details

- StatisticsCollector
- Student
- Paygate
- Notification
- Employee
- N/A
- Instructor
- Manager
- Payment
- TownOffice
- Vehicle
- TheoreticalLesson

- PracticalLesson
- N/A
- Course
- Motocycle
- Car
- Application

Activity Diagram

GetExamResults

Time	Town Office

ExamPeriodFinished

Access town office

[unaccessible]

Inform manager

[accessible]

Prepare table of exam results

Download exam results into database

Notify users about their results

Summary

Name	Description
Swimlane	
Time	
Town Office	
ExamPeriodFinished	
Access town office	
Inform manager	
Prepare table of exam results	
Download exam results into database	
Notify users about their results	

Details

⊞ Swimlane

Activity Partitions
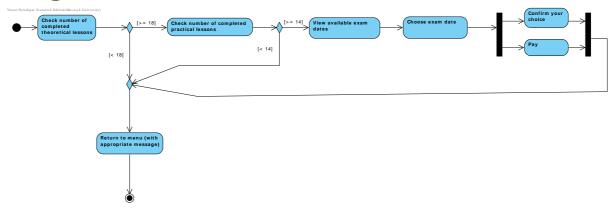
Time		
Contained Elements	**Name**	**Description**
	▦ Access town office	
	● N/A	
	⇶ N/A	
	▦ Inform manager	
	◉ N/A	
	▦ Download exam results into database	
	▦ Notify users about their results	
	◉ N/A	
	⊱ ExamPeriodFinished	

Town Office		
Contained Elements	**Name**	**Description**
	▦ Prepare table of exam results	

● N/A

⊱ ExamPeriodFinished

▦ Access town office

⇶ N/A

▦ Inform manager

◉ N/A

▦ Prepare table of exam results

▦ Download exam results into database

▦ Notify users about their results

◉ N/A

LogIntoTheoretical

Summary
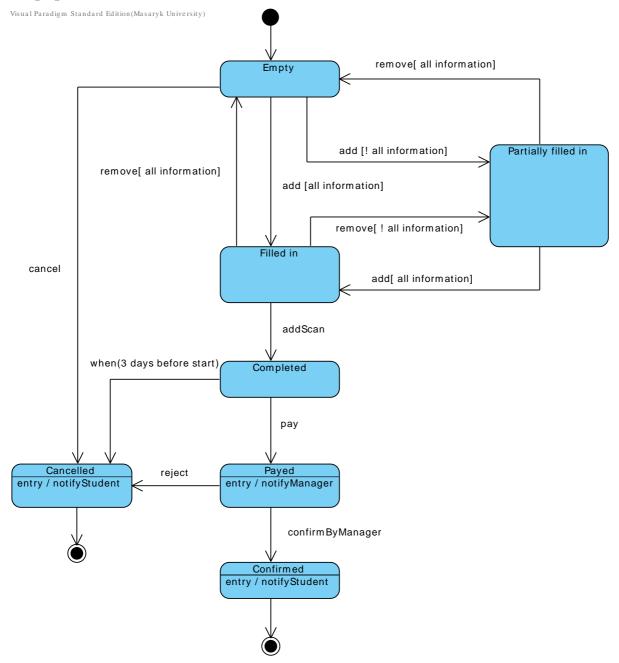
Name	Description
▭ Check number of completed theoretical lessons	
▭ Check number of completed practical lessons	
▭ View available exam dates	
▭ Choose exam date	
▭ Confirm your choice	
▭ Pay	
▭ Return to menu (with appropriate message)	

Details

● N/A

▭ Check number of completed theoretical lessons

⇶ N/A

▭ Check number of completed practical lessons

⇶ N/A

▭ View available exam dates

▭ Choose exam date

N/A

Confirm your choice

Pay

N/A

N/A

Return to menu (with appropriate message)

N/A

Application

Summary

Name	Description
🔲 Empty	
🔲 Filled in	
🔲 Completed	
🔲 Payed	
🔲 Confirmed	
🔲 Cancelled	

Partially filled in	

Details

⬤ N/A

▭ Empty

▭ Filled in

▭ Completed

▭ Payed

Regions

Name	Description
▭ notifyManager	

▭ Confirmed

Regions
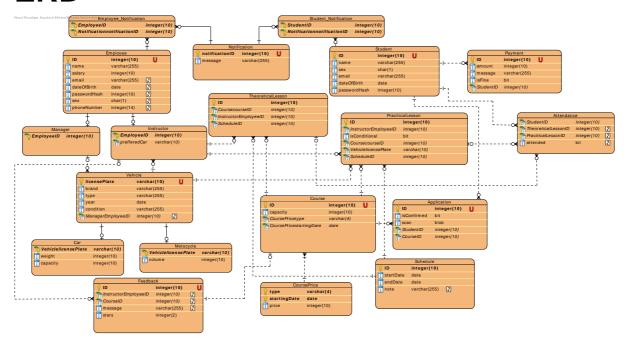
Name	Description
▭ notifyStudent	

◉ N/A

▭ Cancelled

◉ N/A

▭ Partially filled in

Entity Relationship Diagram
ERD

Summary
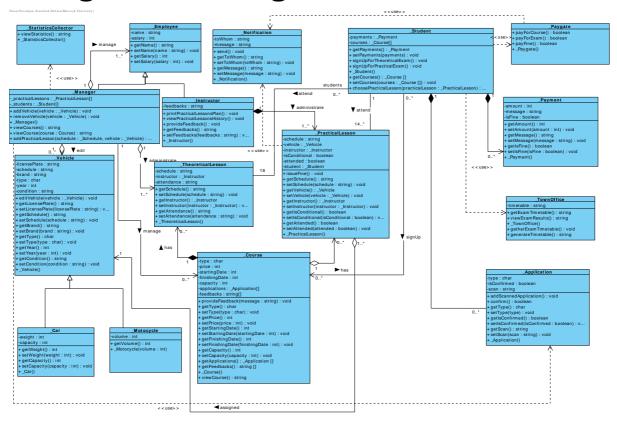
Name	Description
Employee_Notification	
Student_Notification	
Notification	
Student	
Employee	
Payment	
TheoreticalLesson	
Attendance	
PraciticalLesson	
Instructor	
Manager	
Vehicle	
Course	
Application	
Car	
Motocycle	
Schedule	
Feedback	
CoursePrice	

Details

- Employee_Notification
- Student_Notification
- Notification
- Student
- Employee
- Payment
- TheoreticalLesson
- Attendance
- PraciticalLesson
- Instructor
- Manager
- Vehicle
- Course
- Application
- Car
- Motocycle
- Schedule
- Feedback
- CoursePrice
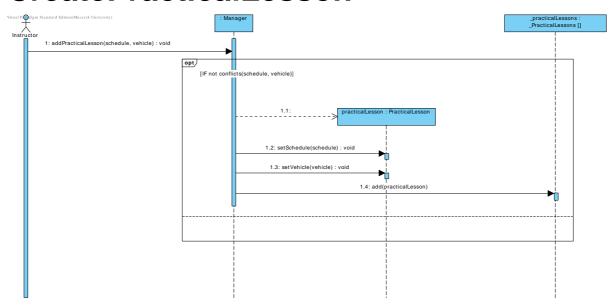
Class Diagram
Design Class Diagram

Summary

Name	Description
_Employee	
_Paygate	
_StatisticsCollector	
_Student	
_Notification	
_Manager	
_Payment	
_Instructor	
_PracticalLesson	
_Vehicle	
_TheoreticalLesson	
_TownOffice	
_Course	
_Application	
_Car	
_Motocycle	

Details

- _Employee
- _Paygate
- _StatisticsCollector
- _Student
- _Notification
- N/A
- _Manager
- _Payment
- _Instructor
- _PracticalLesson
- _Vehicle
- _TheoreticalLesson
- _TownOffice

- _Course
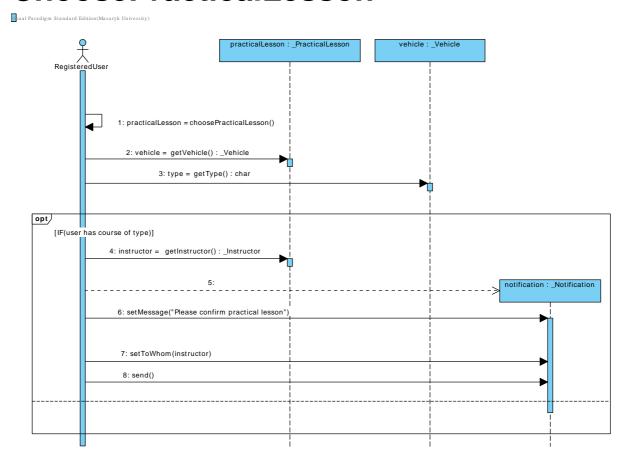- _Application
- N/A
- _Car
- _Motocycle

CreatePracticalLesson

Visual Paradigm Standard Edition(Masaryk University)

Instructor

: Manager

_practicalLessons :
_PracticalLessons []

1: addPracticalLesson(schedule, vehicle) : void

opt

[IF not conflicts(schedule, vehicle)]

1.1:

practicalLesson : PracticalLesson

1.2: setSchedule(schedule) : void

1.3: setVehicle(vehicle) : void

1.4: add(practicalLesson)

Summary

Name	Description
Instructor	
: Manager	
practicalLesson : PracticalLesson	
_practicalLessons : _PracticalLessons []	
IF not conflicts	
Operand2	
alt	

Details

Instructor

: Manager

practicalLesson : PracticalLesson

_practicalLessons : _PracticalLessons []

IF not conflicts

Operand2

alt

ChoosePracticalLesson

Visual Paradigm Standard Edition(Masaryk University)

Summary

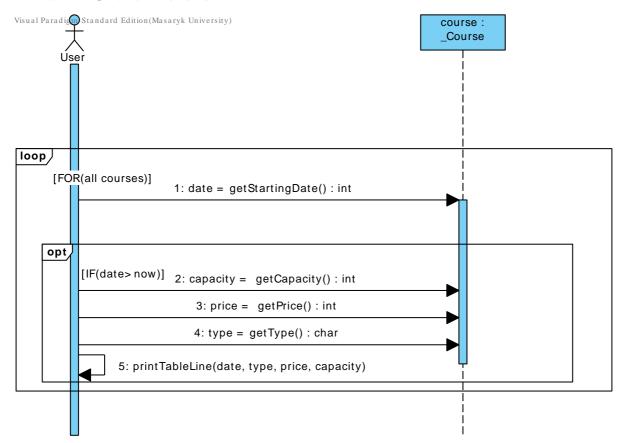| Name	Description
RegisteredUser	
practicalLesson : _PracticalLesson	
vehicle : _Vehicle	
notification : _Notification	
Operand4	
Operand5	
IF(user has course of type)	

Details

RegisteredUser

practicalLesson : _PracticalLesson

vehicle : _Vehicle

notification : _Notification

Operand4

Operand5

IF(user has course of type)

ViewCourses

Summary

Name	Description
☆ User	
⬚ course : _Course	
⬚ Operand3	
⬚ IF(date>now)	
⬚ Operand	
⬚ FOR(all courses)	

Details

☆ User

⬚ course : _Course

⬚ Operand3

⬚ IF(date>now)

Operand

Children
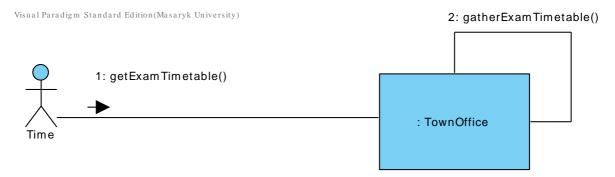
Name	Description
IF(date>now)	

FOR(all courses)

Communication Diagram Get Exam Dates

2: gatherExamTimetable()

1: getExamTimetable()

Time

: TownOffice

Summary
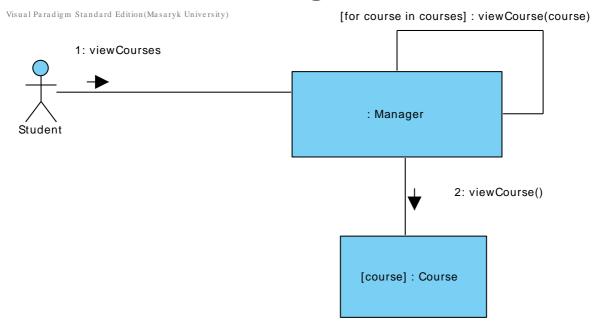
Name	Description
👤 Time	
↝ getExamTimetable()	
▬ : TownOffice	

Details

👤 Time

↝ getExamTimetable()

⊤ : TownOffice

Communication Diagram

Communication Diagram View Courses

[for course in courses] : viewCourse(course)

1: viewCourses

Student

: Manager

2: viewCourse()

[course] : Course

Summary

Name	Description
Student	
viewCourses	
: Manager	
viewCourse()	
[course] : Course	

Details

Student

viewCourses

: Manager

viewCourse()

[course] : Course