NOT part of this diagramm:
WebShells

User
long userId
String firstName
String lastName
Date birthday (deleted)
String password
String mailAddress
Date membershipStart

UserImplementation
long userId

Developer
String organization

APIKey
long keyId
String key
Date expirationDate

ApiKeyDTO
String mailAddress
String key

StudentDTO
long userId
String firstName
String lastName
String mailAddress
int ects

located at
0..*
1

Major
long majorId
String academicTitle
int studentNumber

studies
1
0..*

Student
int ects
boolean isPremiumUser
Collection<Grade> grades

Lecturer
String title

Schedule (external object)
long scheduleId

DeveloperDTO
String mailAddress
String organization

is responsible for
1
0..*

is dean of
0..1
1

Module
long moduleId
String name
int ects
int creditHours
String description

takes
1
0..*

TimeDuration (Embeddable)
long duration
long hours
long minutes
long seconds

ModuleDTO
long moduleId
String name
int ects
int creditHours

Faculty
long facultyId
String name

1
0..*

TimeDurationDTO
long duration

TimeInvest
long timeInvestId

ModuleResults
long moduleResultsId

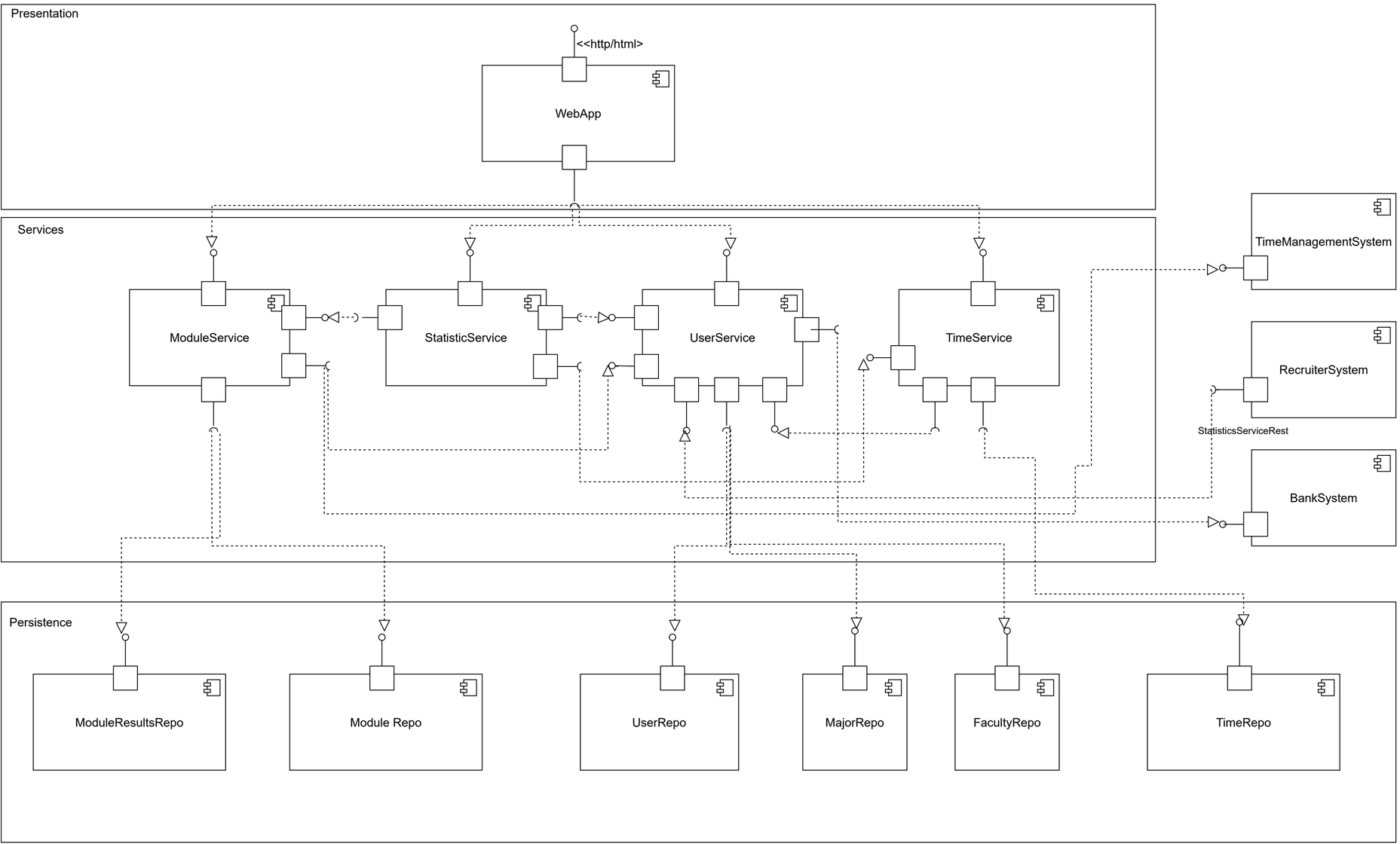
Grade
long gradeId
double value
String description
int tryNumber

TimeOrder
long timeOrderId
Date start
Date end

TimeOrderDTO
String userMailAddress
String moduleName
String startString
String endString

makes

GradeDTO
long gradeId
double value
String description
int tryNumber



<<interface>>
StatisticsServiceRest

+ registerNewDeveloper(DeveloperDTO): APIKeyDTO

+ getStudentByMailaddress(APIKeyDTO, mailAddress): StudentDTO

+ getStudentBy(APIKeyDTO, userId): StudentDTO

+ getModuleGradesForStudent(APIKeyDTO, userId): Map<ModuleDTO, GradeDTO>

+ getModuleTimeDurationForStudent(APIKeyDTO, userId): Map<ModuleDTO, TimeDurationDTO>

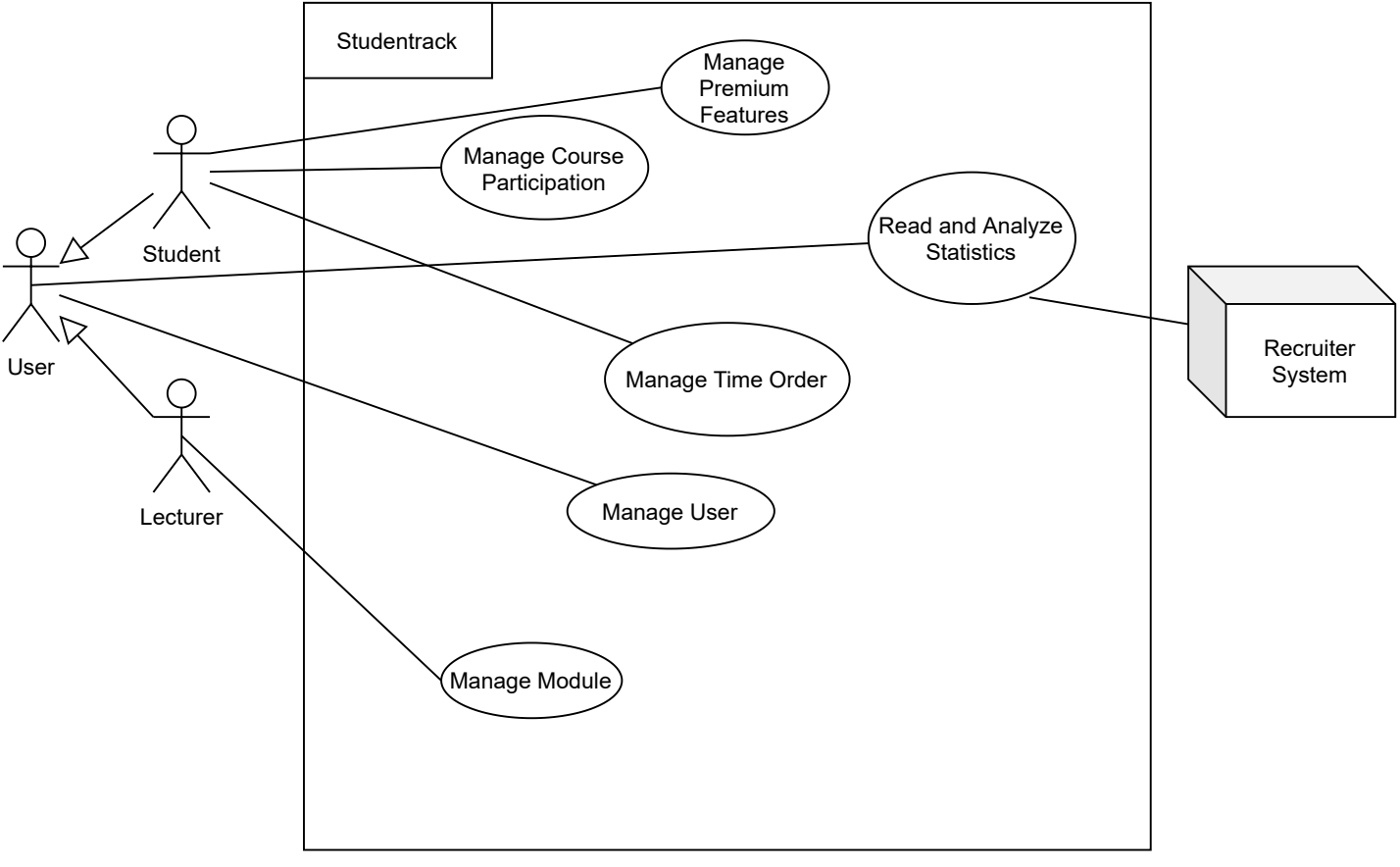
+ getAllModules(APIKeyDTO): Iterable<ModuleDTO>

+ getModule(APIKeyDTO, moduleId): ModuleDTO

+ getAverageGradeForModule(APIKeyDTO, moduleId)

+ getAverageTimeDurationForModule(APIKeyDTO, moduleId)

+ createTimeOrderForStudent(TimeOrderDTO): void



Use Case	Manage User
Pre Conditions	• User is logged in
Description	• User gets formular/page for managing own data • User changes data of their choice (same relevant data as "Register User"), including deletion of account • Validate data • Enter user password for data changes
Post Conditions	• User data is changed • Changes are stored persistently

Use Case	Manage Premium Features
Pre Conditions	• User is logged in • User is student • User has a bank account
Description	• User gets information about the current state of premium features • User enables/disables premium features • User allows/prohibits payment (one time only or as suscription) • Transaction at bank system is triggered • Show confirmation
Post Conditions	• User premium status is updated • Bank is informed • Change are stored persistently

Use Case	Manage Course Participation
Pre Conditions	• User is logged in • User is student • User has no/failed grade in related module for registration OR • User is currently enrolled for course
Description	• User gets formular for course registration • User chooses courses • User confirms the choices for signing in/out • Add/Remove Time Order for User in course • Show Confirmation
Post Conditions	• Course Participation of User is updated • Time Orders are managed according to updates • Changes are stored persistently

Use Case	Read and Analyze Statistics
Pre Conditions	• User is logged in • User has correct permissions for requested statistic depending on User type and premium features
Description	• User chooses statistic of interest (about themself or global anonymized statistics) • User gets data about the requested topic as confirmation
Post Conditions	

Use Case	Manage Time Order
Pre Conditions	• User is logged in • User has access to own (and own course related) Time Orders • User is student
Description	• User gets overview page about own time orders (with options for create, read, change, delete) • User edits start/end/both of Time Order • User confirms changes (if present) • Show confirmation
Post Conditions	• Time Order data is changed (for create, change or delete) • Changes are stored persistently

Use Case	Manage Module
Pre Conditions	• User is logged in • User is lecturer
Description	• User gets overview page about own modules • User edits Module data • User confirms changes (if present) • Show confirmation
Post Conditions	• Module data is changed (for create, change or delete) • Changes are stored persistently