UniDigit

КУРСОВА РАБОТА ПО СОФТУЕРНИ АРХИТЕКТУРИ И РАЗРАБОТКА НА СОФТУЕР

Иванка Паунова 62306
Румен Минчев 62264

Съдържание

1. Въведение

а) Обща информация за текущия документ

- **Предназначение на документа**

Представя софтуерната архитектура на система за управление на процесите и студентската информация в един университет *UniDigit*.

- **Описание на използваните структури на архитектурата.**
  - Декомпозиция на модули - представя цялостен поглед върху системата, разделена на отделни модули, всеки от тях отговарящ за различна функционалност на системата. Основно тя е обособена в 4 главни модула (слоеве) – **Презентационен, Междинен**, **Конфигурация на базата данни** и **База данни**. Всеки от тях комуникира с останалите по строго определен начин, което подобрява производителността и сигурността на системата.
  - Структура на процесите - представя последователността от събития при извършване на процесите в системата **Регистриране но потребител**, **Записване на курс**, **Внасяне на оценка в студентска книжка**, **Подаване на заявка от потребител**. Чрез нея се улеснява промяната и добавянето на нови процеси и модули в система, тъй като дава представа как те комуникират помежду си.
  - Структура на поток на данните - показва потока на информация в системата, представена на различни нива на абстракция. Допълва останалите структури и дава цялостна картина на обработката, достъпа и съхраняването на данни в система. Тя може да бъде в полза на всички заинтересовани лица с цел да се проследи достъпа до данните в системата според типа потребители.

- **Структура на документа**
  - Въведение – секция 1
  - Архитектурни драйвери и обосновка – секция 2
  - Декомпозиция на модули – секция 3
  - Структура на процеси – секция 4
  - Структура на поток на данни – секция 5
  - Архитектурна обосновка – секция 6

б) Общи сведения за системата

UniDigit представлява система за управление на процесите и студентската информация в един университет, която се използва от служителите от Учебен, Административен, Счетоводен отдел, Студентски съвет, преподавателите и студентите в университета. Системата има връзка с външни за нея системи като Държавни публични регистри, Система за управление на учебното съдържание и Система за контрол на НАП и данъчна администрация, с които обменя нужните данни. Тя трябва да осигурява висока защита на личните и финансовите данни от университета, да има висока надеждност и да издържа на пикови натоварвания в моменти на масова употреба на системата (например при записване на изборни дисциплини).

в) Терминологичен речник

- Автентификация - процесът на установяване или потвърждаване истинността на потребителя.
- Архитектурни драйвери - най-влияещите за архитектурата изисквания.
- API - представлява приложно-програмния интерфейс на системата.
- Атака - под атака се разбира злонамерени действия срещу системата с цел сриване на функционалности или неоторизиран достъп.
- База данни - колекция от логически свързани данни в конкретна предметна област, които са структурирани по определен начин.
- Верификация на данните - установяване дали данните са от типа, указан в системата.
- Външна система- отдалечена софтуерна система, която е източник на данни или функционалности, които се използват от настоящата система.
- Coupling - степента, в която един модул разчита на друг модул и е свързан с него. Свързаността може да бъде „висока“ или „ниска“. Ниска свързаност имаме когато един модул не е нужно да се бъде заинтересован от вътрешната имплементация на друг модул.
- Декомпозиция на модули - софтуерна структура, показваща как системата се разделя на отделни модули. Типовете елементи изграждащи тази структура са модули, а връзките между тях са от типа „Х е подмодул на У”.
- Интерфейс (interface) - споделена граница, между която два отделни компонента на системата си обменят информация.
- Контекстна диаграма - диаграма, която дефинира границата между системата или част от системата и нейната среда, показваща обектите, които взаимодействат с нея.
- Конфигурация на базата от данни - начина на разположението на данните в базата данни
- Кохезия - измерител за качеството и степента на взаимодействието на различните компоненти в даден модул.
- Криптиране на данни - замаскирване на данни чрез някакъв филтър с цел тяхното опазване при евентуална успешна атака.
- Междинен слой - в този модул е разположена сървърната част на системата, която осъществява нейната логика. Обособени са различни модули, които представляват отделните функционалности и операции.
- Модул - логически обособена софтуерна единица.
- Надеждност - способността на софтуера да се справя с грешките при изпълнение или способността на алгоритъм да продължи да оперира, независимо от аномалии във входните данни, изчисленията и т.н.
- НАП - Национална агенция по приходите.
- Оторизация - проверка дали някой потребител има право на достъп до някоя функционалност на системата.
- Презентационен слой - Този модул съдържа видовете потребители и са обособени интерфейсите на система, които са различни за съответните типове потребители и осъществят комуникацията между тях и системата.
- Процес - съвкупност от стъпки, която изгражда логическо действие и стига определена цел.

- Софтуерна архитектура - съвкупност от структури, показващи различните софтуерни елементи на системата, външно видимите им свойства и връзките между тях.
- Сриване на функционалност - спирането на дейността на дадена функционалност, поради външна или вътрешна причина.
- Структура на потока от данни - Това е структура, която дава възможност за добиване на ясна представа за информационния поток в системата между нейните компоненти.
- Структура на процесите - софтуерна структура, показваща даден процес през какви условия и действия преминава.
- Сървър - стартирана инстанция на софтуерна система, която може да приема заявки от клиент и да връща подходящи отговори.
- Функционалност - услугата или дейността, която трябва да изпълнява даден модул в системата.

1. Архитектурни драйвери и обосновка

1. Системата обслужва следните отдели в университета:

a. Учебен отдел

b. Счетоводен отдел

c. Студентски съвет

d. Административен отдел

Това изискване определя звената, в които ще се използва системата и то е основно, защото след като се знае къде ще се използва тя, ще могат да се определят изискванията за всеки един отдел. Тези отдели ще бъдат разположени в отделни модули ще се разграничи тяхната функционалност.

2. Всеки отдел предполага наличието на определен тип потребители. Освен това съществуват и администратори на системата, преподаватели и студенти.
Това изискване е важно, защото определя целевите потребители на системата, за които ще могат да се формулират специфичните характеристики и изисквания. Те отново са обвързани с разпределението на модулите в системата. В зависимост от техния тип те ще се обособят различни модули.

3. Системата поддържа профили на студентите и преподавателите, в които се записват техните данни, както и информация за техните компетентности.
Това изискване спомага за определяне на базата от данни на системата. Показва че тя трябва да съхранява данни, като може да се определи тази функционалност да се поддържа отделен модул и ще трябва да се осигури тяхната сигурност.

4. Официалните справки са електронни, като трябва да са защитени от опит за фалшифициране. При желание, справките може да се разпечатват и на хартия, като хартиеното копие трябва да има механизъм за верифициране с електронния вариант на справката.
От това изискване можем да направим извод, че системата поддържа важни данни, до които трябва да се предоставя ограничен достъп и те да се обработват спрямо

потребителите, които ги използват. Трябва да има компоненти в системата, които да осигуряват тяхната сигурност и да управляват достъпа им.

5. Системата да поддържа защита на всички лични и финансови данни от неоторизиран достъп.
Това изискване определя основно качество на системата, свързано с нейната сигурност и поддръжката на данни. Трябва да се осъществяват процеси на оторизация и автентикация на потребителите, които имат достъп до данните в системата.

6. Системата да предоставя API (публичен интерфейс) за достъп до генерираните официални справки и публични събития.
От това изискване се определя един от основните компоненти на системата – API (публичен интерфейс), което е важно за архитектурата. Тя трябва да предоставя улеснено използване от страна на потребителите ѝ.

7. Системата трябва да е достъпна 24/7, като изключение за поддръжка и планирано обновяване се допуска само по време на официални празници.
От това изискване следва, че при проектиране на архитектурата ѝ трябва да се вземе под внимание осигуряването на висока надеждност за сметка на не толкова добра наличност.
С оглед на това, че официалните празници в сферата на образование не са толкова малък процент, можем да направим извод, че тя няма да бъде високо налична.

8. Системата трябва да прави връзка със следните външни системи:

a. Държавни публични регистри за текущи студенти, към които периодично (напр. 2 пъти годишно) се изпраща информация за статуса на студентите. Изпращаната информация се контролира от потребителите от учебен и административен отдел.

b. Система за контрол на национална агенция за приходите и данъчната администрация.

c. Система за управление на учебното съдържание (напр. Moodle, но може и да е друга система, която се употребява в конкретния университет)

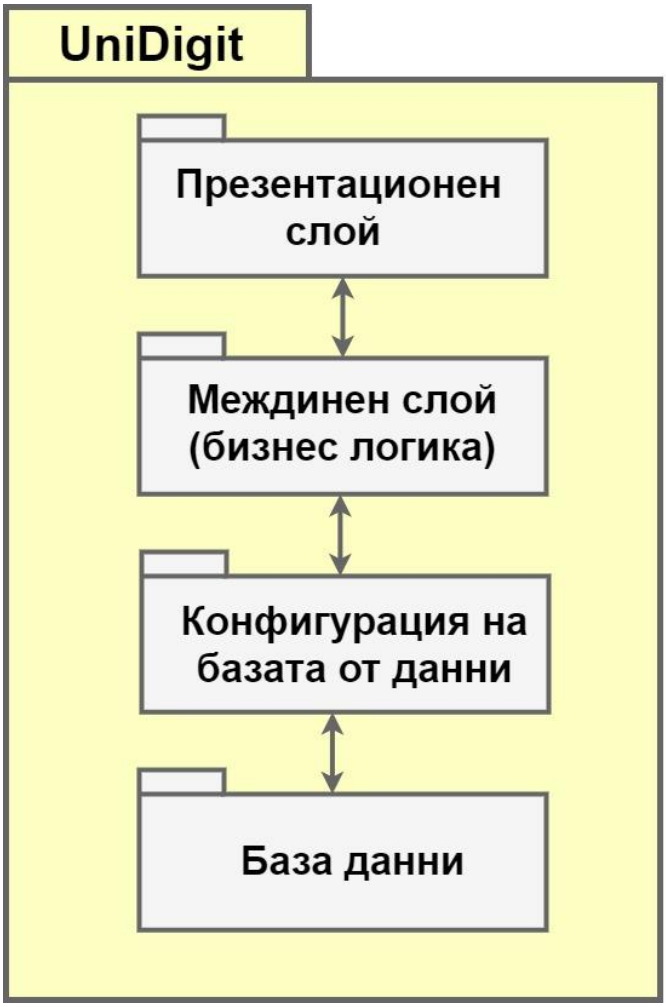
d. Списъкът с външни системи, с които се прави връзка може да се увеличи в процеса на използване на системата.

От това изискване можем да определим с кои външни системи си взаимодейства университетската система, като това е от значение за нейната архитектура, защото представя обвързаността ѝ с тях. Оттук ще се направи извод за потока на информация в системата и как тя се споделя и обмена с външните системи.

9. Системата да може да издържа на пикови натоварвания (например увеличаване на потребителските заявки по време на кампания за записване на изборни дисциплини, вписване на оценки по време на сесия и т.н.), като може да обработва едновременно 1000 заявки в секунда.
При проектиране на системата, основавайки се на това изискване, трябва да се вземе под внимание, че тя трябва да има добра производителност, особено в критични за нея моменти, като споменатите в изискването пикови натоварвания.

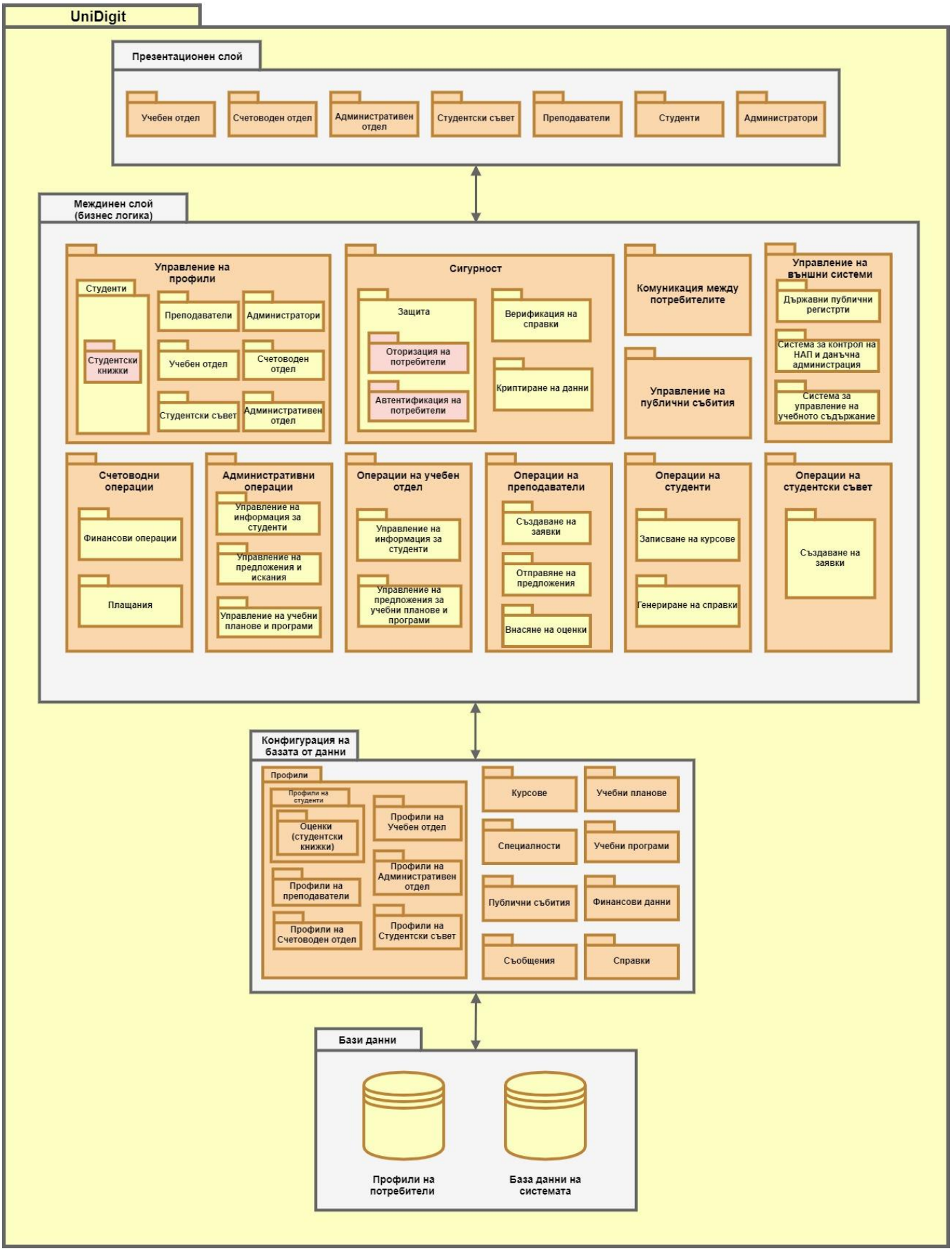
3. Декомпозиция на модулите

а) Общ вид на декомпозицията на модули за системата



Системата *UniDigit* е разделена в 4 основни модула – Презентационен, Междинен, Конфигурация на базата данни и базата данни. Избрахме представянето чрез клиент – сървър архитектурен стил и по-точно трислойна архитектура, защото тя гарантира по-добра производителност и сигурност на една система. За конкретната система това са две от най-важните изисквания, които трябва да бъдат спазени с оглед на това, че трябва тя да поддържа голям брой различни по вид потребители, трябва да издържа на пикови натоварвания от множество заявки и да съхранява в защитен вид лични данни и финансови операции. С разпределянето на функционалностите на системата в отделни независими слоеве ще спомогне да се осъществят исканите качества.

б) Контекстна диаграма



Всеки от отделните главни модули изпълнява определена функция и комуникира с останалите по определен начин. Модулът, който е свързан с потребителите, представя единствено пряката им връзка със системата, като осигурява достъпен интерфейс. Той има връзка с междинния модул, където са разположени операциите и логиката на системата и няма директна връзка с базата от данни, с цел осигуряване на по-добра защита. Междинния модул е свързан с базата от данни посредством модул, който описва нейната конфигурация и детайлизира съдържанието ѝ. По този начин се защитават данните и се постига тяхното управление, като не е разрешен директен достъп от страна на потребителите и данните са независими от тях и логиката на системата. Той представлява преход между потребителите и базите от данни, които съхраняват данните на системата. С разпределянето на системата в различни нива, относно дейностите и отговорностите, които изпълнява в главните модули се постига ниско ниво на coupling и в същото време силна кохезия в отделните модули, което води до високо качество на системата – ефективност и надеждност.

в) Подробно описание на всеки модул

1. Презентационен модул

- **Предназначение на модула**

Този модул съдържа видовете потребители и са обособени интерфейсите на система, които са различни за съответните типове потребители и осъществят комуникацията между тях и системата.
В презентационния модул се съдържат 6 отделни модула – Учебен отдел, Счетоводен отдел, Административен отдел, Студентски съвет, Преподаватели и Студенти. Всеки един от тях представлява потребителски интерфейс предоставен на потребителите в зависимост от техния тип. Чрез него те осъществяват директен достъп и комуникация със самата система.

- **Основни отговорности на модула**

Той предоставя информация и директен достъп до потребителите на системата. Чрез него те се свързват със нея и той е междинната им връзка с операциите и данните в системата. Отделянето на тази връзка в отделен модул осигурява защита на данните, като потребителите са отделени в независим модул, чрез който не могат директно да ги достъпват.

2. Междинен модул (бизнес логика)

- **Предназначение на модула**

В този модул е разположена сървърната част на системата, която осъществява нейната логика. Обособени са различни модули, които представляват отделните функционалности и операции. Той комуникира с презентационния слой, осъществяващ връзката с потребителите, и със слоя за данните, където е разположена базата на системата.

- **Основни отговорности на модула**

Отговорността на този модул се състои в това да съхранява логическата част на системата. Операциите, характерни за всеки от видовете потребители, са разположени в отделни модули, специфицирани по техния тип. Така се улеснява възможността за промяна на системата и добавянето на нови функционалности, като логиката е разделена за различните отдели. Осъществява се и управлението на профилите в системата, както и процеса по управление на достъпа до нея, обособен в модула за сигурност.

2.1. Управление на профили

В този модул е представена отговорността за управлението на профилите в системата, като дава различни права на отделните потребители, имащи профили. Потребителите от тип студенти и преподаватели имат профили в системата за всяко лице, докато останалите потребители от Учебен, Административен, Счетоводен отдел и Студентски съвет имат служебни профили, като е предоставен един профил за всеки отдел, т.е. служителите от тези отдели нямат персонални профили, в които да осъществяват вход, а достъпват профила, отговарящ за техния отдел. В този модул се съдържа и логиката за студентската книжка като част от профила на студента, до която имат достъп и преподавателите. В отделен подмодул са представени и профилите на администраторите, чрез които те получават достъп до операциите по управление и поддържане на системата.

Интерфейси:

- *login (username, password)*
  - входни данни - потребителско име и парола на студент или преподавател
  - изходни данни - успешна или неуспешна операция
  - грешки и изключения – връща съобщение за грешка при некоректни данни
  - извършва се автентификация на потребителите преди осъществяването на входа им в системата
  - зависимости от други елементи - връзка с модула **Студенти** и **Преподаватели** от Презентационния модул и **Автентификация на потребителите** от модула Защита.

- *createProfile (userNumber)*
  - входни данни - уникален идентификационен номер на потребител
  - изходни данни - създаден профил на потребител
  - грешки и изключения – връща съобщение за грешка при съществуващ обект с подадения номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Профили** в базата данни след криптиране

- *viewProfile (userNumber)*
  - входни данни - уникален идентификационен номер на потребител
  - изходни данни - информация за потребител
  - грешки и изключения – връща съобщение за грешка при несъществуващ обект с такъв номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Профили** в базата данни..

- *modifyProfile (userNumber, newData)*
  - входни данни - уникален идентификационен номер на потребител и променени данни
  - изходни данни - обновен профил на потребител
  - грешки и изключения – връща съобщение за грешка при несъществуващ обект с такъв номер или невалидни данни
  - зависимости от други елементи – получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Профили** в базата данни.

- *createCourse (course, data)*
  - входни данни - уникален идентификационен номер на курс и данни за него
  - изходни данни - отговор за успешно или не създаден курс
  - грешки и изключения - връща съобщение за грешка при вече съществуващ курс с подадения номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Курсове** в базата данни

- *createSpecialty (specialty, data)*
  - входни данни - уникален идентификационен номер на специалност и данни за него
  - изходни данни - отговор за успешно или не създадена специалност
  - грешки и изключения – връща съобщение за грешка при вече съществуваща специалност с подадения номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Специалности** в базата данни

- *updateCourse (course, newData)*
  - входни данни - уникален идентификационен номер на курс и променени
  - изходни данни – обновен курс
  - грешки и изключения - връща съобщение за грешка при несъществуващ курс с подадения номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Курсове** в базата данни

- *updateSpecialty (specialty, newData)*
  - входни данни - уникален идентификационен номер на специалност и променени данни
  - изходни данни - обновена специалност
  - грешки и изключения - връща съобщение за грешка при несъществуваща специалност с подадения номер
  - зависимости от други елементи - получава заявката от **Администратори** от Презентационния слой и я изпраща към модула **Специалности** в базата данни

- *viewGrades (course)*
  - входни данни: курс, за който да се видят оценките

- изходни данни: оценки за съответния курс
- грешки и изключения: съобщение за грешка при несъществуващ курс или незаписан студент в съответния курс
- зависимости от други елементи - получава заявката от **Студенти** от презентационния слой и я изпраща към модула **Оценки** в базата данни

2.2. Сигурност

Тук се осъществяват операциите по сигурността и защитата на системата. Съдържа модул за верификация на данните, който е необходим при генерирането на различни справки от страна на студентите. Разположен е и механизмът за криптиране на данни, който се използва за предпазването от изтичане на данни и за осигуряване на по-добра сигурност. Тук се съдържа и една от основните части за осигуряване на защита на системата, а именно авторизацията и оторизацията на потребителите, чрез които се извършва устояването на атаки към системата. Това е необходимо, за да се контроли достъпът им до личните и финансовите данни в системата.

Интерфейси:

- *verifyUser (username, password)*
  - входни данни - потребителско име и парола на студент или преподавател
  - изходни данни - коректност на данните
  - зависимости от други елементи - заявката се изпраща към модула **Профили** в базата данни

- *encrypt (data)*
  - входни данни - данни за криптиране
  - изходни данни - криптиран вид на данните

- *verifyReport (reportNumber)*
  - входни данни - уникален идентификационен номер на справка
  - изходни данни – отговор за успешна или неуспешна верификация
  - зависимости от други елементи – получава заявката от модула **Генериране на справки** от Операции на студенти и я изпраща към модула **Справки** в базата данни

2.3. Комуникация между потребителите

Това е логиката, която изпълнява изискването за осигуряване на възможност за изпращане на лични съобщения между потребителите. В него е представен процесът, чрез който те си комуникират.

Интерфейси:

- *sendMessage (userNumber, message)*
  - входни данни - идентификационен номер на потребител, на когото ще се изпраща съобщението и съответното съобщение
  - изходни данни - отговор за успешен или не процес на изпращане
  - зависимости от други елементи - връзка с всички модули от Презентационния слой и изпраща заявката към модула **Съобщения** от базата данни след криптиране

- *receiveMessage (userNumber, message)*

- входни данни - идентификационен номер на потребител, който е изпратил съобщението и самото съобщение
- изходни данни - отговор за успешен или не процес на получаване
- зависимости от други елементи - връзка с всички модули от Презентационния слой и изпраща заявката към модула **Съобщения** от базата данни

2.4. Управление на публични събития

Този модул е отговорен за възможността на потребителите за създаване на публични събития и управление на информацията, свързана с тях. Обособен е като отделен модул в логиката на система, понеже всички потребители могат да осъществяват тази функционалност

Интерфейси:

- *createEvent (description)*
  - входни данни - информация за събитие
  - изходни данни - уникален идентификационен номер на събитие
  - грешки и изключения - връща съобщение при неуспешно създаване на събитие
  - зависимости от други елементи - връзка с всички модули от Презентационния слой и изпраща заявката към модула **Събития** от базата данни

- *modifyEvent (eventNumber, newData)*
  - входни данни - идентификационен номер на събитие и променени данни
  - изходни данни - отговор за успешна или не промяна на събитието
  - грешки и изключения - връща съобщение при несъществуващо събитие с такъв номер или некоректни данни
  - зависимости от други елементи - връзка с всички модули от Презентационния слой и изпраща заявката към модула **Събития** от базата данни

2.5. Управление на външни системи

Това е модулът, който осъществява връзката между *UniDigit* и външни за нея системи. Решението за съществуването на този модул в логиката на системата е повлияно от изискването за възможност за добавяне на нови външни системи, които да работят с конкретната. Съществуването на отделен модул, който съдържа тази информация, значително улеснява изискваната функционалност за изменяемост на системата, а именно добавянето на нови външни системи, което ще се осъществи значително по-лесно когато логиката и процесите за комуникация с тях са обособени в отделен модул.

2.6. Счетоводни операции

Този модул показва процесите, свързани с извършване на операции от страна на потребителите от счетоводния отдел, които управляват финансовите данни, чрез извършване на финансови операции и разплащания.

2.7. Административни операции

Тук са разположени операциите на административния отдел, отделени в модули съответно за управление на информацията на студентите, който е отговорен за връзката с външната система за Държавни публични регистри, към която трябва да се предоставя информация за студентите, и за управление за предложения и искания, отправени от страна на потребителите от тип преподаватели, Студентски съвет и Учебен отдел.

Интерфейси:

- *manipulateStudentInformation (studentNumber, newData)*
  - входни данни - идентификационен студентски номер и променени данни
  - изходни данни - отговор за успешна или не модификация
  - грешки и изключения - връща съобщение при невалиден студентски номер, т.е. несъществуващ студент
  - зависимости от други елементи - получава заявката от модула **Административен отдел** от Презентационния слой

- *processOffer (offerNumber, userNumber)*
  - входни данни - уникален номер на предложение и идентификационен номер на потребител, който го е изпратил
  - изходни данни - отговор за прието или не предложение
  - грешки и изключения - връща съобщение при несъществуващо предложение (некоректен номер)
  - зависимости от други елементи – получава заявката от модула **Административен отдел** от Презентационния слой и я изпраща след одобрение към модула **Учебни планове** или **Учебни програми** от базата данни

- *createProgram (programNumber, data)*
  - входни данни - уникален идентификационен номер на учебен план и данни за него
  - изходни данни - отговор за успешен или не процес на създаване
  - грешки и изключения - връща съобщение за грешка при вече съществуваща учебна програма с подадения номер
  - зависимости от други елементи - получава заявката от **Административен отдел** от Презентационния слой и я изпраща към модула **Учебни програми** от базата данни

- *updateProgram (programNumber, newData)*
  - входни данни - уникален идентификационен номер на учебна програма и променени данни
  - изходни данни - отговор за успешен или не процес на промяна
  - грешки и изключения – връща съобщение за грешка при несъществуваща учебна програма с подадения номер
  - зависимости от други елементи - получава заявката от **Административен отдел** от Презентационния слой и я изпраща към модула **Учебни програми** от базата данни

- *createPlan (planNumber, data)*

- входни данни - уникален идентификационен номер на учебен план и данни за него
- изходни данни - отговор за успешен или не процес на създаване
- грешки и изключения - връща съобщение за грешка при вече съществуващ учебен план с подадения номер
- зависимости от други елементи - получава заявката от **Административен отдел** от Презентационния слой и я изпраща към модула **Учебни планове** от базата данни

- *updatePlan (planNumber, newdata)*
  - входни данни - уникален идентификационен номер на учебен план и променени данни
  - изходни данни - отговор за успешен или не процес на промяна
  - грешки и изключения - връща съобщение за грешка при несъществуващ учебен план с подадения номер
  - зависимости от други елементи - получава заявката от **Административен отдел** от Презентационния слой и я изпраща към модула **Учебни планове** от базата данни

2.8. Операции на Учебен отдел

Тук е логиката свързана с процесите, свързани със служителите в Учебен отдел. Те са отделени в модули, съответно за управление на информация за студенти, който отново е отговорен за връзката с Държавните публични регистри за предоставяне на информация за студенти и управление за предложения за учебни планове и програми, получени от страна на преподавателите.

Интерфейси:

- *manipulateStudentInformation (studentNumber, data)*
  - входни данни - идентификационен студентски номер и данни за промяна
  - изходни данни - отговор за успешна или не модификация
  - грешки и изключения - връща съобщение при невалиден студентски номер, т.е. несъществуващ студент
  - зависимости от други елементи - получава заявката от модула **Учебен отдел** от Презентационния слой

- *processOffer (offerNumber, userNumber)*
  - входни данни - уникален номер на предложение и идентификационен номер на потребител, който го е изпратил
  - изходни данни - отговор за прието или не предложение
  - грешки и изключения - връща съобщение при несъществуващо предложение (некоректен номер)
  - зависимости от други елементи - получава заявката от модула **Учебен отдел** от Презентационния слой и има връзка с **Курсове** и **Специалности** от базата данни, за да се получи информация за кои от тях е направено предложението

2.9. Операции на студенти

Тук са представени процесите, чрез която потребителите от тип студенти ще могат да записват курсове и да генерират на справки от различен характер за студентския им

статус. Тези функционалности отново са разделени в отделни подмодули, отговорни за съответната функционалност.

Интерфейси:

- *enrollCourse (courseNumber)*
  - входни данни - идентификационен номер на курс за записване
  - изходни данни - отговор за успешно или не записване на курс
  - грешки и изключения - връща съобщение за грешка при невалиден курс или при неудовлетворяване на изискванията за съответния курс
  - зависимости от други елементи – получава заявката от модула **Студенти** от Презентационния слой и има връзка с модулите **Профили на студенти** и **Курсове** от базата данни.

- *getReport (type)*
  - входни данни - вид на справка
  - изходни данни - генерирана справка
  - грешки и изключения - връща съобщение за грешка при некоректен тип на справката
  - зависимости от други елементи - получава заявката от модула **Студенти** от презентационния слой и има връзка с модулите **Профили на студенти** и **Справки** от базата данни.

2.10. Операции на преподаватели

Преподавателите имат функциите в система, представени в модулите за управление на оценки на студентите в студентските им книжки, създаването на заявки за различни искания към Административния отдел и отправяне на предложения за учебни планове и програми към Учебен отдел.

Интерфейси:

- *createQuery (description)*
  - входни данни - описание на заявката за искане
  - изходни данни - уникален идентификационен номер на създадена заявка
  - зависимости от други елементи - получава заявката от модула **Преподаватели** от Презентационния слой и я изпраща към модула **Управление на предложения и искания** от Административни операции.

- *sendOffer (specialtyNumber, courseNumber, description)*
  - входни данни – идентификационни номера на специалност и курс за предложението и неговото описание
  - изходни данни - уникален номер на изпратеното предложение
  - грешки и изключения – връща съобщение за грешка при невалидни курс или специалност
  - зависимости от други елементи - получава заявката от модула **Преподаватели** от Презентационния слой и има връзка със **Специалности** и **Курсове** от базата данни и **Управление на учебни планове и програми** от Операции на Учебен отдел

2.11. Операции на Студентски съвет

Потребителите от студентския съвет могат единствено да създават заявки към Административен отдел за различни искания. Логиката на тази операция е представена в подмодула за създаване на заявки, който е отговорен за този процес.

Интерфейси:

- *createQuery (description)*
  - входни данни - описание на заявката за искане
  - изходни данни - уникален идентификационен номер на създадена заявка
  - зависимости от други елементи - получава заявката от модула **Студентски съвет** от Презентационния слой и я изпраща към модула **Управление на предложения и искания** от Административни операции

3. Конфигурация на базата от данни

Този модул представлява описание на елементите, които ще се съдържат в базата от данни. Той се грижи за това да достъпва базата, да чете и да връща информация от нея. Отделните подмодули представляват типовете данни, които ще се съхраняват. Те са: данни за потребителите и профилите им в системата, данни за специалности, курсове, учебни планове и програми, данни от студентските книжки, представени като оценки, финансови данни, ползвани от Счетоводния отдел, съобщения, изпратени и получени между потребителите и данни за направените справки от потребителите, които имат достъп до тази функционалност. Наличието на този модул представлява още една степен на защита на данните.

Интерфейси:

- *registerUser (userNumber, data)*
  - входни данни - уникален идентификационен номер на потребител и данни
  - изходни данни - регистриран потребител
  - грешки и изключения - съобщение за грешка при съществуващ запис с въведения номер
  - зависимости от други елементи - получава заявката от модула **Управление на профили** в Междинния слой и я изпраща към базата от данни за профилите на потребителите

- *getUser (userNumber)*
  - входни данни - уникален идентификационен номер на потребител
  - изходни данни - информация за потребител
  - грешки и изключения - връща съобщение за грешка при несъществуващ обект с такъв номер или невалидни данни
  - зависимости от други елементи - получава заявката от модула **Управление на профили** в Междинния слой и я изпраща към базата от данни за профилите на потребителите

- *updateUser (userNumber, newData)*
  - входни данни - уникален идентификационен номер на потребител и променени данни

  - изходни данни - обновен профил на потребител
  - грешки и изключения - връща съобщение за грешка при несъществуващ обект с такъв номер или невалидни данни
  - зависимости от други елементи - получава заявката от модула **Управление на профили** в Междинния слой и я изпраща към базата от данни за профилите на потребителите

- *createCourse (coureNumber, data)*
  - входни данни - уникален идентификационен номер на курс и данни за него
  - изходни данни - отговор за успешен или не процес на създаване
  - грешки и изключения - връща съобщение за грешка при вече съществуващ курс с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getCourse (courseNumber)*
  - входни данни - уникален идентификационен номер на курс
  - изходни данни - информация за курс
  - грешки и изключения - връща съобщение за грешка при несъществуващ курс с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *updateCourse (courseNumber, newData)*
  - входни данни - уникален идентификационен номер на курс и променени данни
  - изходни данни - обновена информация на курс
  - грешки и изключения – връща съобщение за грешка при несъществуващ курс с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createSpecialty (specialtyNumber, data)*
  - входни данни - уникален идентификационен номер на специалност и данни за него
  - изходни данни - отговор за успешен или не процес на създаване
  - грешки и изключения - връща съобщение за грешка при вече съществуваща специалност с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getSpecialty (specialtyNumber)*
  - входни данни - уникален идентификационен номер на специалност
  - изходни данни - информация за специалност
  - грешки и изключения - връща съобщение за грешка при несъществуваща специалност с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *updateSpecialty (specialtyNumber, newData)*
  - входни данни - уникален идентификационен номер на специалност и променени данни
  - изходни данни - обновена информация за специалност
  - грешки и изключения - връща съобщение за грешка при несъществуваща специалност с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createPlan (planNumber, data)*
  - входни данни - уникален идентификационен номер на план и данни за него
  - изходни данни - отговор за успешен или не процес на създаване
  - грешки и изключения - връща съобщение за грешка при вече съществуващ план с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getPlan (planNumber)*
  - входни данни - уникален идентификационен номер на учебен план
  - изходни данни - информация за учебен план
  - грешки и изключения - връща съобщение за грешка при несъществуваща учебен план с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *updatePlan (planNumber, data)*
  - входни данни - уникален идентификационен номер на план и променени данни
  - изходни данни - обновена информация на учебен план
  - грешки и изключения - връща съобщение за грешка при несъществуващ план с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createProgram (programNumber, data)*
  - входни данни - уникален идентификационен номер на учебна програма и данни за нея
  - изходни данни - отговор за успешен или не процес на създаване
  - грешки и изключения - връща съобщение за грешка при вече съществуваща учебна програма с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getProgram (programNumber)*
  - входни данни - уникален идентификационен номер на учебна програма
  - изходни данни - информация за учебна програма
  - грешки и изключения - връща съобщение за грешка при несъществуваща учебна програма с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *updateProgram (programNumber, data)*
  - входни данни - уникален идентификационен номер на учебна програма и променени данни
  - изходни данни - обновена информация на учебна програма
  - грешки и изключения - връща съобщение за грешка при несъществуващ план с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *setGrades (studentNumber, grade, data)*
  - входни данни - идентификационен студентски номер, оценка и идентификационен номер на курс
  - изходни данни - отговор за успешен или не процес на нанасяне на оценка
  - грешки и изключения - връща съобщение за грешка при несъществуващ студент с подадения номер или несъществуващ курс
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getGrades (studentNumber, courseNumber)*
  - входни данни - идентификационен студентски номер и идентификационен номер на курс
  - изходни данни - оценки на студент по определен курс
  - грешки и изключения - връща съобщение за грешка при несъществуващ студент с подадения номер или несъществуващ курс
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createMessage (sender, receiver, message)*
  - входни данни – идентификационни номера на подател, получател, съобщение
  - изходни данни - записано съобщение
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getMessages (sender, receiver)*
  - входни данни - идентификационни номера на подател и получател
  - изходни данни - съобщения между подател и получател
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createEvent (eventNumber, description)*
  - входни данни - идентификационен номер на събитие и описание
  - изходни данни - успешен или не процес на създаване на събитие
  - грешки и изключения - връща съобщение за грешка при вече съществуващо събитие с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getEvent (eventNumber)*
  - входни данни - идентификационен номер на събитие
  - изходни данни - информация за събитие
  - грешки и изключения - връща съобщение за грешка при несъществуващо събитие с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *updateEvent (eventNumber, newData)*
  - входни данни - идентификационен номер на събитие, променени данни
  - изходни данни - обновена информация за събитие
  - грешки и изключения - връща съобщение за грешка при несъществуващо събитие с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *createReport (reportNumber, data)*
  - входни данни - идентификационен номер на справка и данни на справката
  - изходни данни - успешен или не процес на създаване на справка
  - грешки и изключения - връща съобщение за грешка при вече съществуваща справка с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

- *getReport (reportNumber)*
  - входни данни - идентификационен номер на справка
  - изходни данни - информация за справка
  - грешки и изключения - връща съобщение за грешка при несъществуваща справка с подадения номер
  - зависимости от други елементи - изпраща заявката към базата от данни на системата

4. Бази от данни (слой за данните)

Това е модулът, който се грижи за данните на системата. Те се разделени в две отделни бази – едната, която съдържа само информация за профилите на потребителите и другата, в която са останалите данни на системата. Разделението на данните в две отделни бази е в следствие от изискването за производителност и защита на системата. По този начин в критични моменти, при опити за атаки към системата, свързани с вход към нея и евентуалното сриване на функционалността за вход в система и достъпване до данните в базата за профилите на потребителите, то останалата част от нея ще продължи цялостната си работа, като останалите модули няма да бъдат засегнати. Разделението на данните е още едно ниво на сигурност, тъй като при проникване в данните от едната база, то няма да се наруши целостта на другите.

д) Описание на възможните вариации

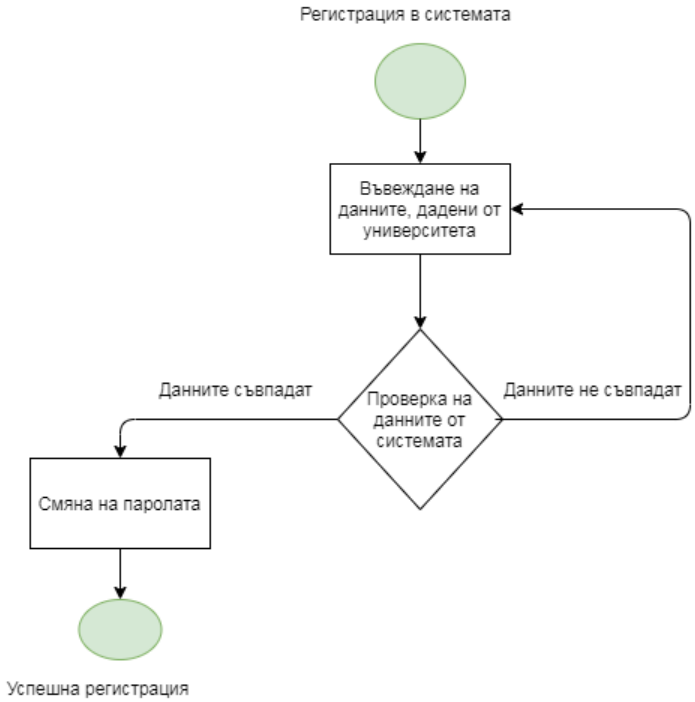
Към модула Управление на външни системи могат да се добавят външни изпълнители на услуги за извършване на операциите по разплащания от страна потребителите от Счетоводен отдел.

4. Структура на процесите

- **Мотивация за избор**

Избрахме тази структура, защото имаме много процеси в нашата система и би било добре те да бъдат обяснени по ясен и достъпен начин. Това ще се случи най-добре чрез структура на процесите.

- **Процес на регистрация на потребител**



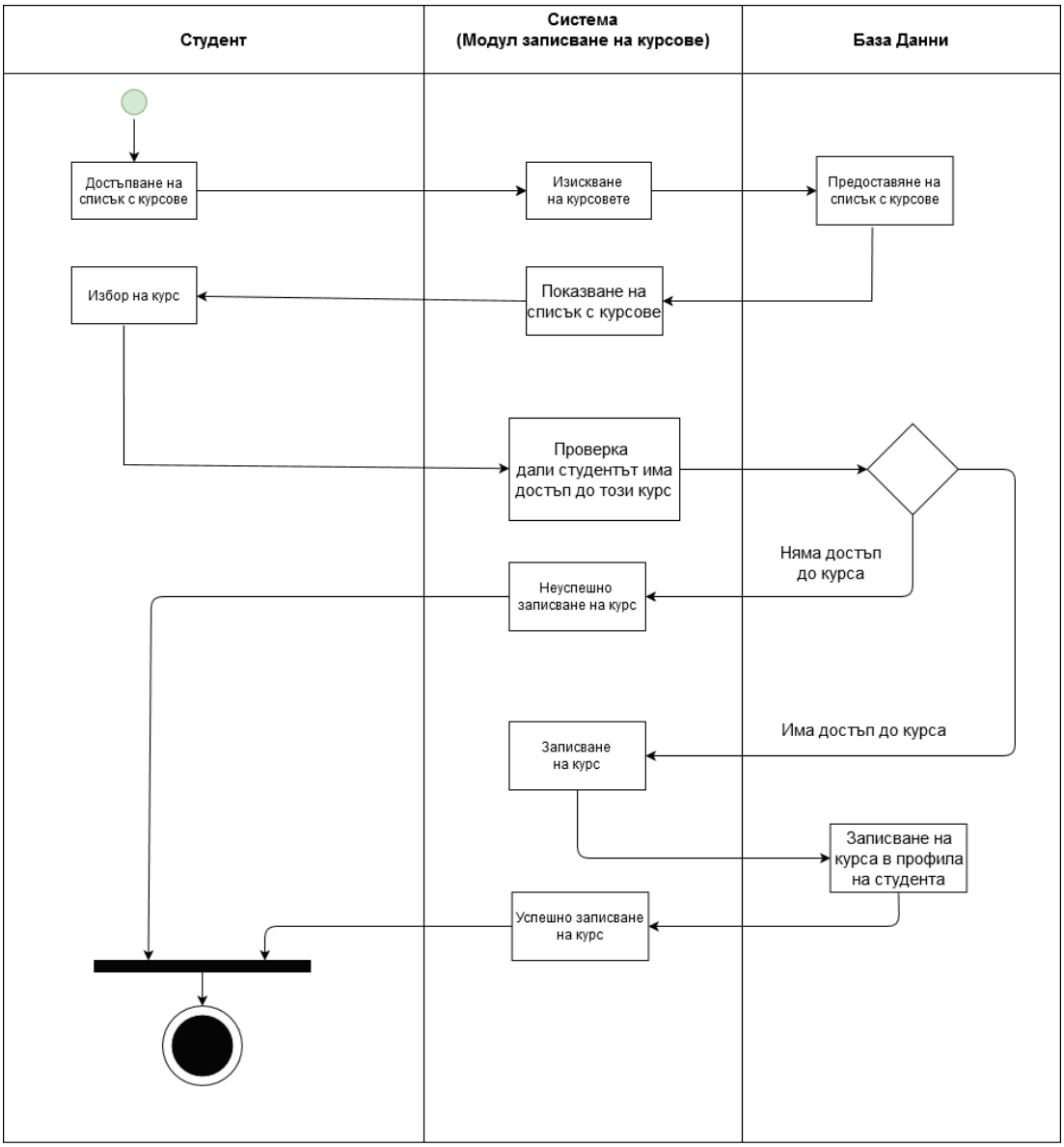
- **Описание на елементите и връзките**

Регистрирането на потребител – студент или преподавател, става чрез попълването на данни дадени от университета. Те представляват уникално име и парола. Потребителят въвежда тези данни и системата проверява дали съвпадат като търси в модула база данни за потребители. Ако данните не съвпаднат, се дава нов шанс на потребителя да ги въведе, а ако съвпаднат той трябва да промени служебната парола дадена му от университета с някоя, която е удобна за него. Тази нова парола се записва в базата от данни за този потребител и той вече може да се възползва от функционалностите на системата, до които има достъп.

- **Описание на обкръжението**

Регистрирането на потребител става чрез връзка с Интернет и предоставени данни от университета.

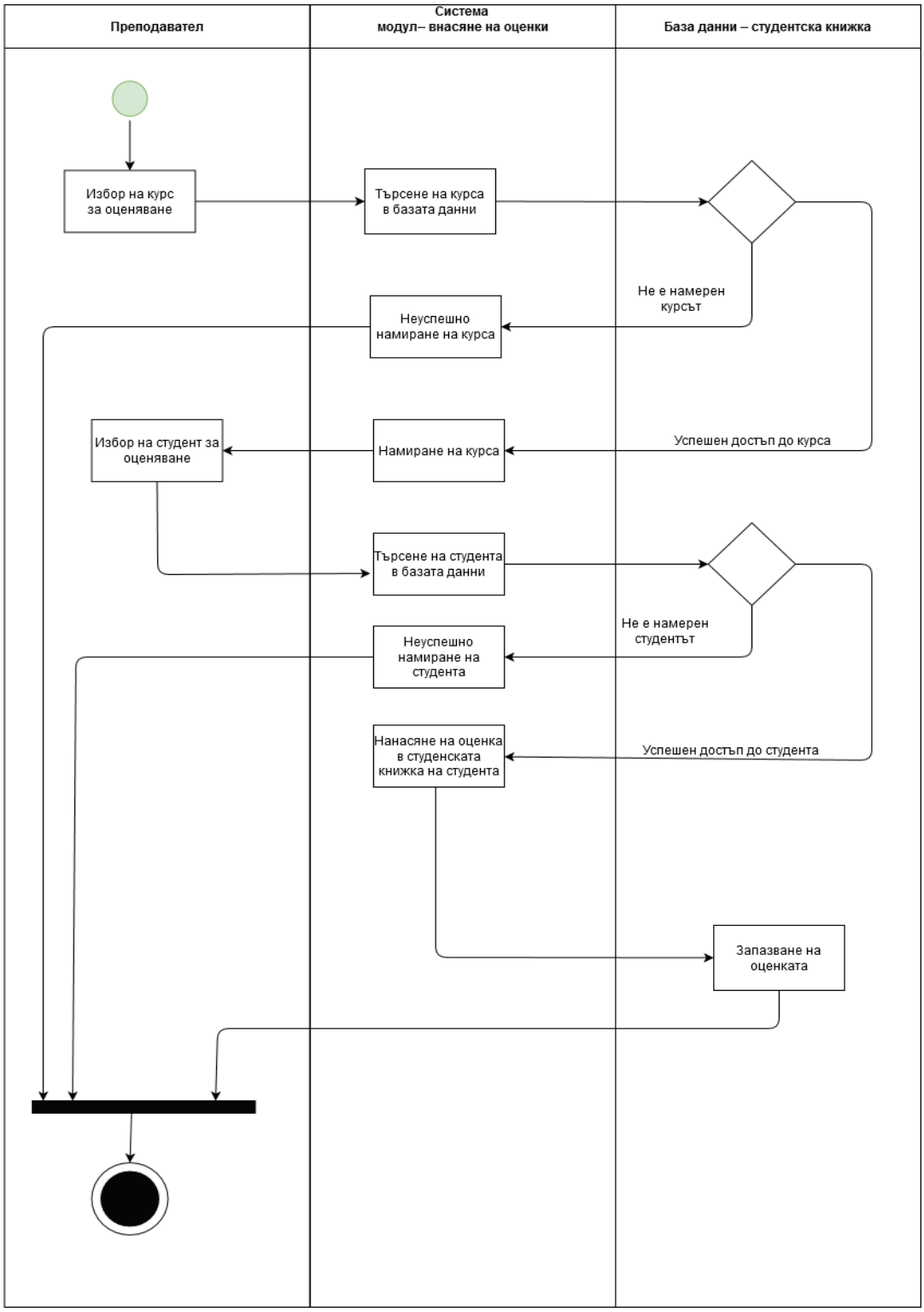
- **Процес на записване на курс**



- **Описание на елементите и връзките**

Записването на курс от потребител тип студент става единствено като студентът достъпи списъка с предмети и натисне бутон за записване на предмета, който си е избрал. След това системата проверява изискванията за този предмет и го записва, ако те са изпълнени, иначе не го записва. Модулите, които участват в този процес са Студенти, Операции на студенти или по-точно казано Записване на курс и База данни, откъдето се взима информацията за изискванията за всеки курс. Проверката дали студентът може да запише курсът се намира в базата от данни.

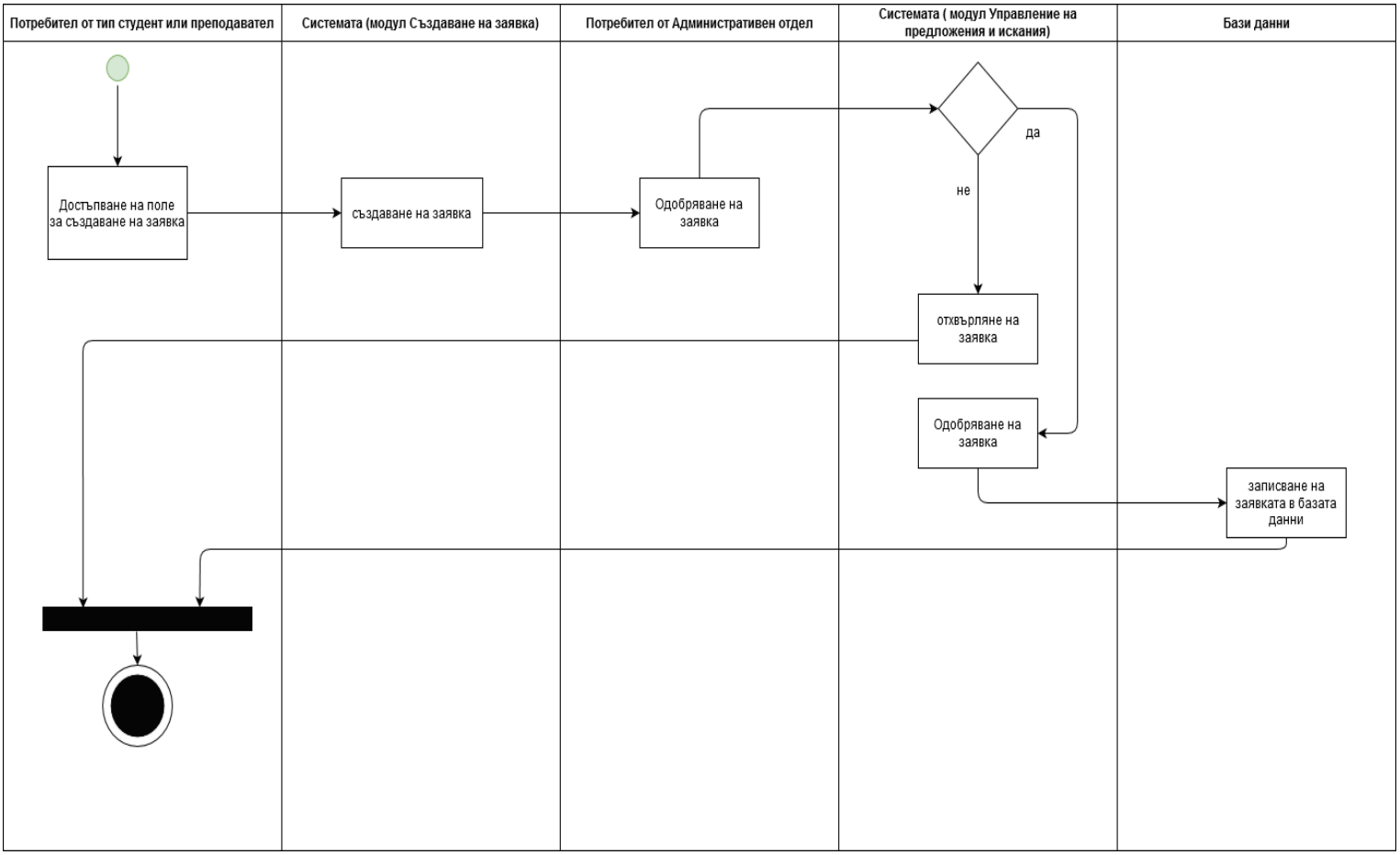
- **Процес на нанасяне на оценка в студентската книжка на студент**



- **Описание на елементите и връзките**

Нанасянето на оценка от потребител от тип преподавател става след като успешно преподавателят намери предмета, по който преподава, и студента, на когото преподава, в базата данни. След тези действия той може да запише оценката в неговата студентска книжка. Модулите, които участват тук са Преподавател, Студент и База данни.

- **Процес на подаване на заявка от потребител в системата**



- **Описание на елементите и връзките**

Заявка може да се създаде само от потребител от тип студент или преподавател. След създаването заявката трябва да бъде одобрена от Административния отдел и записана в базата данни, чрез уникален номер. На по-късен етап заявката бива разглеждана и обработена от потребителите, които имат такава отговорност.

5. Структура на потока на данни

- **Мотивация за избор**

UniDigit е система, в която главна роля имат данните – тяхното съхраняване, обработване и трансферът им представляват част от функциите, които са използвани в основните процеси на системата. **Структурата на потока на данните** ще даде възможност за добиване на ясна представа за информационния поток в системата между нейните компоненти.

Множеството разновидности на потребители определя и големия обмен на данни със системата и между отделните потребители. От **декомпозицията на модулите** не става ясно как се разпределят данните между отделните потребители и модули на системата. Затова **структурата на потока на данните** ще визуализира този процес. Направен е изборът за представяне на **логическа структура на поток на данните** срещу физическа, тъй като тя ще представи процесите и информацията в системата на бизнес ниво. По този начин тя е идеален инструмент за комуникация с потребителите на системата. Те ще могат да добият представа кои данни от кой потребител се обработват и съответно кой потребител има достъп до различните видове данни, които протичат в системата. Тази структура ще бъде от полза и в случай на включването на нов потребител в системата – той ще може да се запознае с информационния поток, което значително ще улесни работата му. Обменът на информация между голям брой видове потребители на система, както е и случаят в *UniDigit*, може да бъде неясен и дори объркващ. Наличността на структура на поток на данните ще осигури визуализиране на информационния поток и ще изясни обмена на информация между отделните потребители.

Външните системи, с който работи университетската система, отново са точки, с който тя обменя данни. От **декомпозицията на модулите** не става ясен техният обмен. **Структурата на поток на данните** ще даде представа за потока на данните между външните системи и *UniDigit*, като би била и от полза на потребителите на отделните външни системи да получат информация за това, кой отговаря за данните, които те получават и при проблем да могат да се ориентират правилно към съответния отдел, имащ връзка конкретно към данните, които се обменят с нея. Може да бъде подпомогнато осигуряването на изискването за възможност за увеличаване броя на системите, с които се прави връзка, в процеса на използването на системата, освен със **структура на процесите**, но и с наличността на **структура на поток на данните**. Тя ще даде възможност за проследяване на потока на информация и ще спомогне за интеграцията на външните системи по правилен начин.

Съхраняването на данни, които имат важно значение за един университет, а именно данни за студенти, курсове, дейности на университета, изисква тяхната защита, което е и едно от изискванията на системата. Освен от **декомпозицията на модулите**, където са обособени отделни модули отговарящи за осигуряване на това изискване, то може да бъде подпомогнато и от **структура на поток на данните**. Тя ще даде ясна представа за това къде се използват данните в системата и съответно кой потребител ги използва, което ще допринесе за улесняване на контролирането на достъпа до данните от страна на отделните потребители на системата.

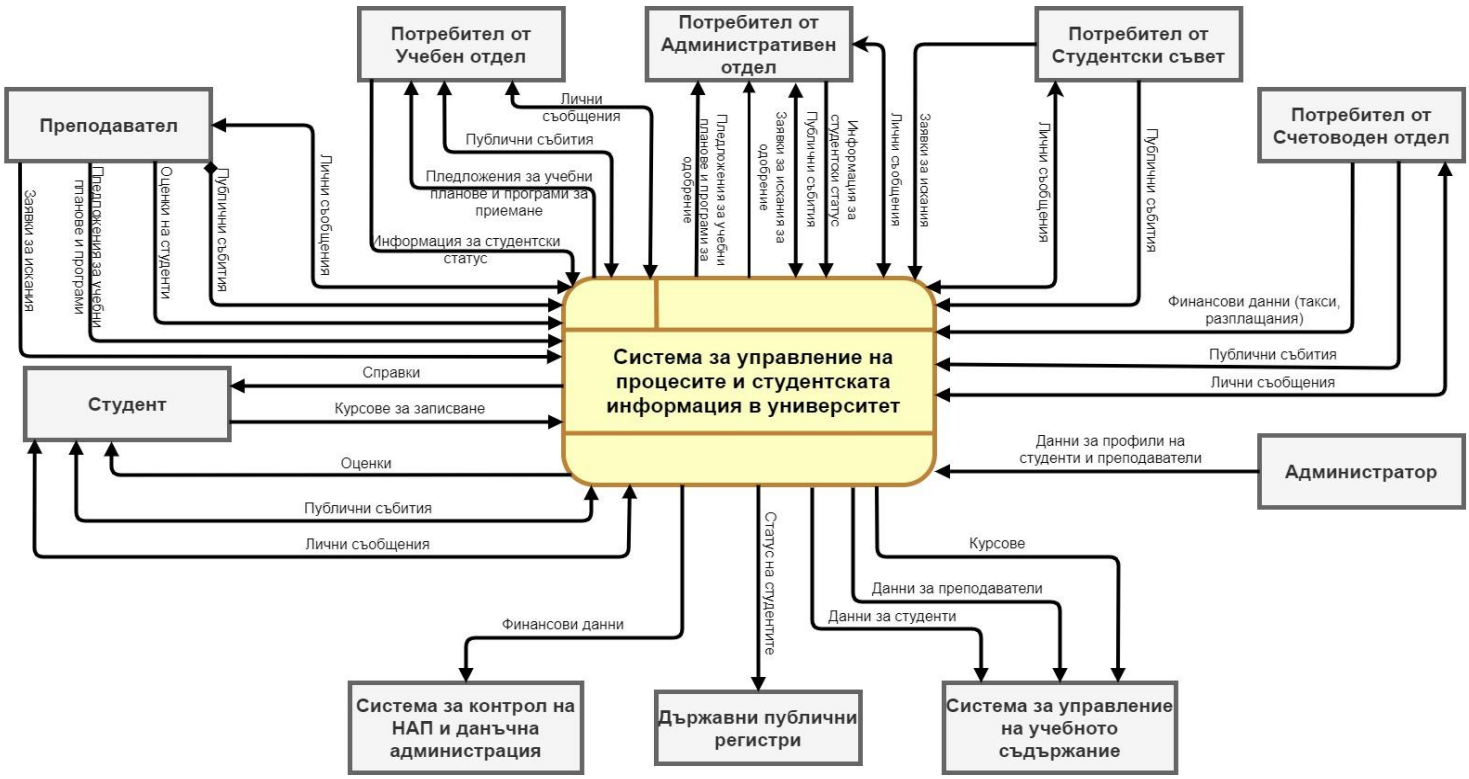
Изискването за висока надеждност на системата не е достатъчно осигурено от **декомпозицията на модулите**. **Структура на поток на данните** има отношение по въпросите на бързодействието по време на изпълнението и високата надеждност на системата. Правилното разпределение да данните, както и използването от страна на потребителите само на необходимите за техните дейности данни ще подобри работата на системата с оглед на нейното бързодействие и надеждност, като не протича излишно количество данни при изпълнението на определен процес в системата и се поддържа тяхната коректност, като се предоставя достъп до тях от потребителите, които имат нужните права.

Структурата на поток на данните представлява допълнение към **декомпозицията на модулите** и **структурата на процесите**, като описва прехвърлянето на данни между процесите в системата и показва тяхното движение. Визуализира обмена на данните, като може да се направи извод за това кои модули какви данни използват и потока им между процесите в системата.

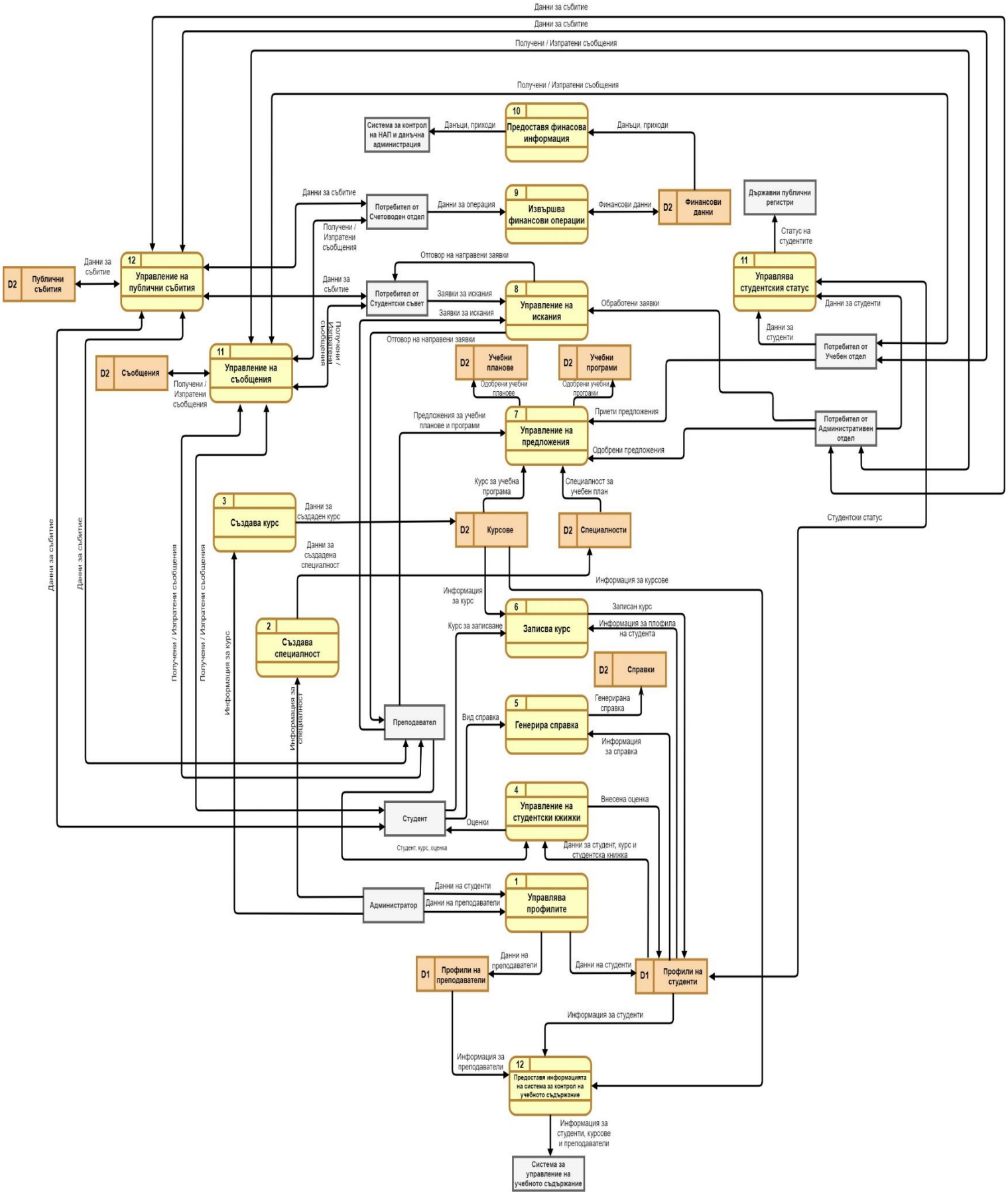
Тази структура представя функциите и процесите, които съхраняват, обработват и разпространяват данни между системата и нейното обкръжение, като позволява преглед на различни нива на детайлност. Стартира от общ поглед върху данните на системата, като постепенно може той да се разширява в по-подробно описани диаграми на различни нива на абстракция.

- **Първично представяне - ниво 0**

Диаграмата показва данните на системата на фундаментално ниво. Определя достъпа на отделните потребители и външни системи до данните на **UniDigit** и дава информация чрез кои данни си комуникират помежду си.



- **Представяне - ниво 1**

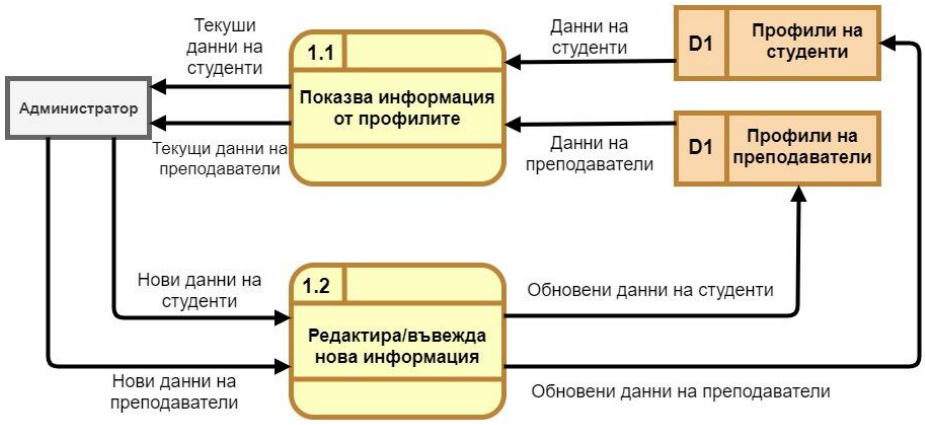


Тази диаграма по-детайлно описва потока на данни в системата. Показани са основните процеси и се добива представа за това кой процес кои данни използва, от кой го получава и с кой комуникира. Представя и съхранението на данните на системата.

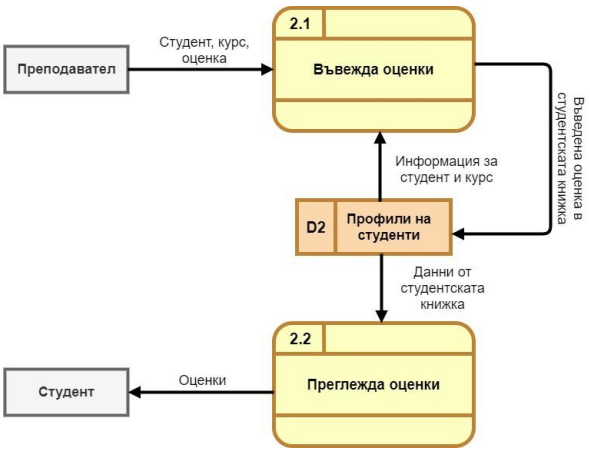
- **Представяне - ниво 2**

На това ниво са представени някои от процесите в ниво 1, разделени на подпроцеси и по-детайлно описан поток на данни. Дава още по-ясна визуализация на предоставянето, използването, преработването, съхраняването на данните на системата.

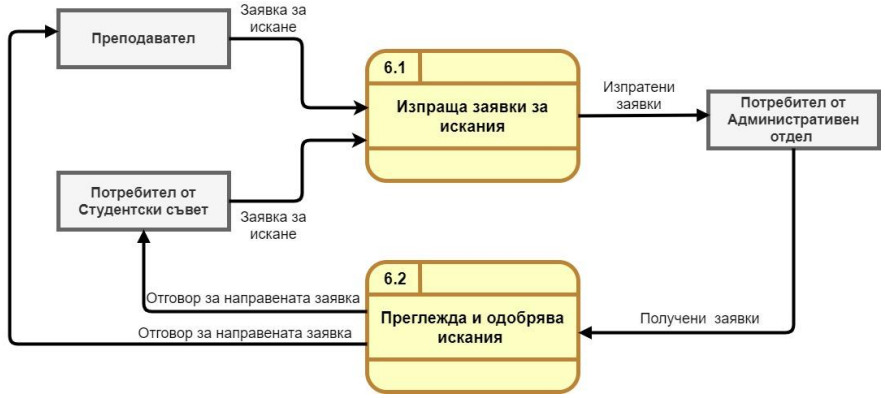
- Управление на профилите



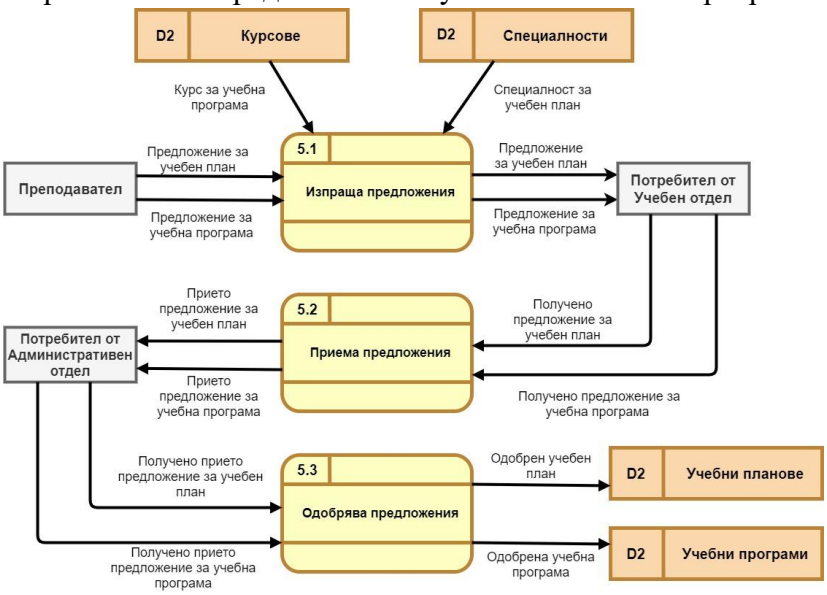
- Управление на студентски книжки



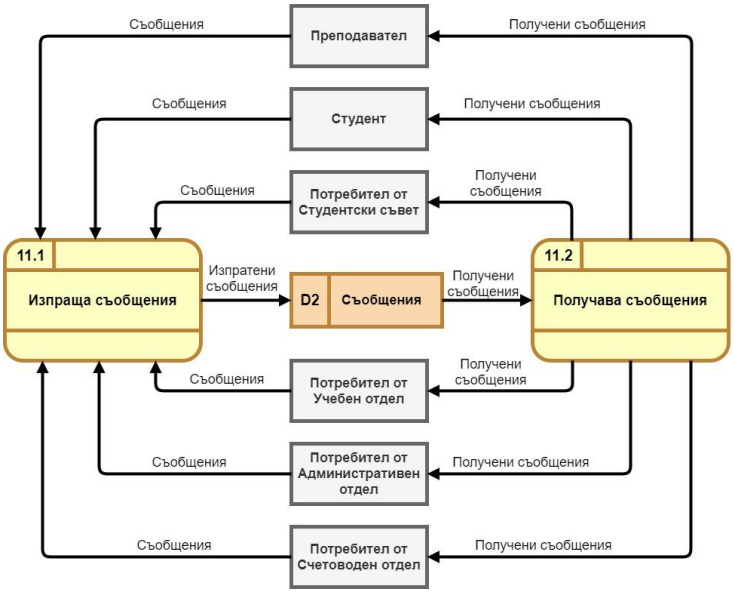
- Управление на заявки за искания



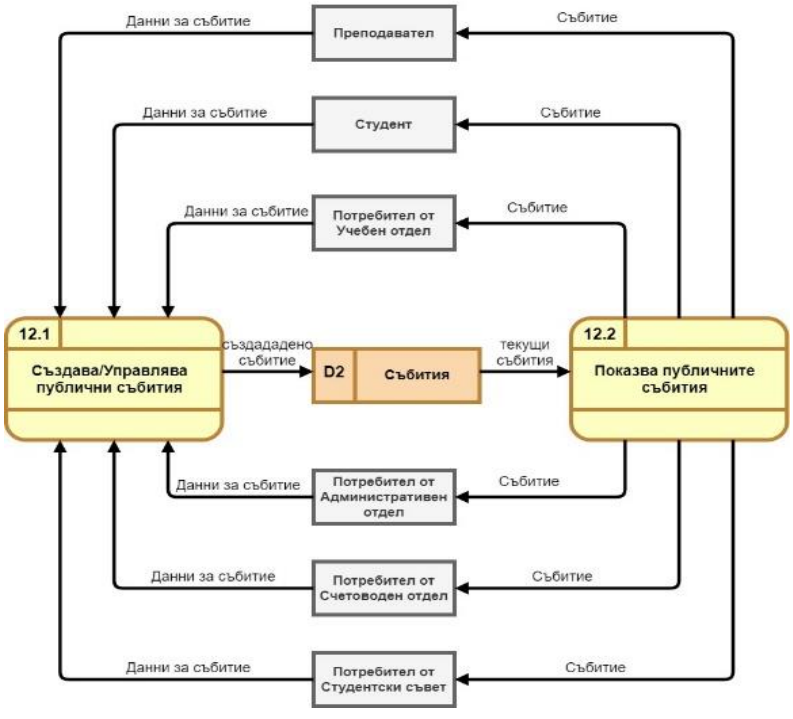
- Управления на предложения за учебни планове и програми



- Управление на съобщения



- Управление на публични събития



- **Описание на елементите и връзките**

Структурата на поток на данни представя елементите като потребители и външни системи за *UniDigit*, които обменят информация с нея, посредством процеси осигуряващи връзката с външните източници и данните на системата.

Източници, които предоставят или получават данни от системата:

- Администратор - предоставя данни за профилите на потребителите от тип преподаватели и студенти и може да получава съответно информация за тези профили, както и да достъпва и въвежда информация за курсовете и специалностите в университета.
- Преподавател - източник за системата на данни като предложения за учебни планове и програми, заявки за искания, оценки на студенти.
- Студент - получава информация от системата, свързана със студентската книжка, а именно оценки, както и различни справки за студентския си статус и предоставя като данни курс за записване.
- Потребител от Учебен отдел - обменя данни за направени предложения за учебни планове и програми, както и данни свързани със студентския статус
- Потребител от Административен отдел - обменя данни за направени предложения за учебни планове и програми, направени заявки за различни искания, както и данни свързани със студентския статус
- Потребител от Студентски съвет - източник за системата на данни, представляващи заявки за искания
- Потребител от Счетоводен отдел - обменя данни със системата, свързани с финансовите операции (такси, плащания и други)
- Държавни публични регистри - получават данни, свързани със статуса на студентите
- Система за управление на учебното съдържание - получават данни за курсовете, студентите и преподавателите
- Система за контрол на НАП и данъчна администрация – получават данни за приходите и данъците на университета и служителите

Процеси, осъществяващи потока на данни в системата:

- Система за управление на процесите и студентската информация в университет - включва всички процеси в системата, които осъществяват потока на данните, обменът им между потребителите и съхранението им в и извличането им от базата данни.
- Управлява профили - приема данни за преподаватели и студенти от администратора и му предоставя данни за съответните потребители от базата данни на системата.
- Управление на студентски книжки - включва процесите Въвежда и Преглежда оценки, които съхраняват данни от преподавателите в студентските книжки и предоставят данни от студентските книжки на студентите.
- Генерира справка - по получен вид справка от студент, генерира данни от профила на студента и съхранява справката в базата данни, където тя може да бъде достъпена евентуално за верификация.

- Записва курс - получава курс от студент, данни за курса и профила на студента от базата данни, при съвпадение в изискванията осъществява записване и обновява информацията в профила на студента.
- Управление на предложения за учебни планове и програми - включва процесите Изпраща, Приема, Одобрява предложение, чрез които се обменят данни за предложения между Преподаватели, Учебен отдел и Административен отдел. Достъпва данни за курсове и специалности от базата данни и съхранява в нея такива за одобрени планове и програми.
- Управление на заявки за искания - включва процесите Изпраща заявки за искания и Преглежда и одобрява заявки за искания, чрез които обмена данни за искания между Преподаватели, Студентски съвет и Административен отдел.
- Извършва финансови операции - приема финансови данни от Потребители от Счетоводен отдел, които обработва и съхранява в базата от данни и получава от нея необходима информация за целите на операциите.
- Предоставя финансова информация - трансферира финансови данни като приходи и данъци от базата данни към Система за контрол на НАП и данъчна администрация.
- Управлява студентски статус - получава данни от потребители от Административен и Учебен отдел, както и от профилите на студентите от базата данни, където и съхранява обновена информация. Предоставя данни за студентския статус, получени от профилите на студентите към Държавни публични регистри.
- Предоставя информация на система за контрол на учебното съдържание - трансферира данни за курсове, студенти и преподаватели от базата данни към Система за управление на учебното съдържание.
- Управление на съобщения - включва процеси Изпращане и Получаване на съобщение, които обменят данни между всички видове потребители на системата и съхраняват и извличат съобщения от базата данни.
- Управление на публични събития - включва процеси Създава/Управлява публични събития и Показва публични събития, които осъществяват трансфер на данни за събития между всички видове потребители на системата и ги съхраняват и получават от базата данни.

Хранилища на данни:

- Профили на студенти - предоставят процеси свързани с управлението на студентската информация като Управление на профили, Управление на студентски книжки, Генериране на справки, Записване на курс, Управление на студентски статус и съхраняват данни от процеси като Управление на профили, Управление на студентски статус, Управление на студентски книжки и записване на курс. Това хранилище е в отделна база, съдържаща информация за профилите в системата.
- Профили на преподаватели - предоставя и съхранява информация от процес управление на профилите. Това хранилище е в отделна база, съдържаща информация за профилите в системата.
- Курсове - предоставя информация на процеси Записване на курс, Управление на предложения, Предоставяне на информация на система за контрол на учебното съдържание и съхранява данни от процес на Създаване на курс.

- Специалности - предоставя информация на процес Управление на предложения и съхранява информация от процес Създаване на специалност.
- Учебни планове - съхранява данни за одобрени предложения за програми от процес Управление на предложения.
- Учебни програми - съхранява данни за одобрени предложения за планове от процес Управление на предложения
- Финансови данни - предоставят и данни на процеси свързани с финансови операции и предоставяне на финансови данни на Система за контрол на НАП и данъчна администрация и съхраняват данни от извършени финансови операции.
- Справки – съхранява информация от процес Генериране на справка.
- Съобщения - съхраняват данни от процес Изпращане на съобщение и предоставят данни на процес Получаване на съобщение.
- Събития - съхраняват данни от процес Създаване на публични събития и предоставят данни на процес Показване на публични събития.

- **Описание на обкръжението**

Данните се кешират от страна на сървъра (междинния слой), за да се подобри производителността на системата.

- **Описание на възможни вариации**

Към процеса Управление на профили могат да се включат и хранилища за останалите потребители на системата в които да се съхраняват данни за тях, предоставени от Администратор и да се извличат данни, които да се получават от него с възможност за модификация и управление от негова страна. По този начин той ще може да достъпва и обмена със системата данни, свързани с всички потребители и ще може да се създаде процес за вход освен на студенти и преподаватели и на останалите потребители, който ще получава от тях данни за вход, като ще ги верифицира с тези от базата данни, съхраняваща данни от профилите им и ще осъществява вход в системата. Това ще създаде още едно ниво на защита на системата, освен това осигурено с различните интерфейси на всеки вид потребител, предоставящи им само достъпните за тях данни и операции.

Администраторът може да има и достъп до съобщенията в системата, като има възможност също да изпраща и получава такива от останалите потребители с цел информиране за определени събития или промени, които са настъпили или трябва да се направят от негова страна.

6. Архитектурна обосновка

1. Системата обслужва следните отдели в университета:
а) Учебен отдел
б) Счетоводен отдел
в) Студентски съвет
г) Административен отдел

Това изискване е реализирано чрез **Декомпозицията на модулите**, като за всеки отдел е обособен отделен модул отговарящ за интерфейса на системата в презентационния слой и такъв за техните специфични функционалности в сървърната част (Междинния слой).

2. Всеки отдел предполага наличието на определен тип потребители. Освен това съществуват и администратори на системата, преподаватели и студенти.

Подобно на предишното изискване е реализирано и това, като в допълнение са обособени модули за *Преподаватели, Студенти* и *Администратори* в *Презентационния и Междинния слой*.

3. Потребителите от учебен отдел приемат предложения за учебни планове и програми от преподавателите.

4. Предложенията за учебни планове (специалности) и програми (курсове) се одобряват от административния отдел.

Най-ясна представа за това изискване се дава от *Декомпозицията на модулите*, където има отделни модули изпълняващи тези операции *Управление на предложения* в *Административни операции* и *Операции на Учебен отдел* и *Отправяне на предложения* в *Операции на преподаватели*. Проследяването на данните за предложенията за учебни планове и програми се осъществява от *Структурата на поток на данните* като се показва по достъпен начин кои потребители достъпват тези данни.

5. Системата поддържа профили на студентите и преподавателите, в които се записват техните данни, както и информация за техните компетентности.

В модула *Конфигурацията на базата данни* са обособени модули за профилите на студенти и преподаватели, които са отговорни за тяхното съхранение. Може да се види кой има достъп до тези данни и как те преминават през системата от *Структура на потока на данните* и как се управляват и биват използвани от *Структура на процесите.*

6. Потребителите от счетоводния отдел, контролират финансовите операции, които засягат другите потребители (студентски такси, възнаграждения на служителите, и др.), както и разплащания с външни изпълнители на услуги.

В *Декомпозицията на модулите* е представен отделен модул за интерфейса на *Счетоводния отдел*, както и такъв за операциите извършвани от него. Данните които се обработват от тези операции и се достъпват от счетоводния отдел са представени в *Структурата на потока на данните.*

7. Студентите могат да се записват одобрени курсове, само в рамките на тяхната специалност и при условие, че профилът им отговаря на входните изисквания за компетентности за съответния курс.

Тази функционалност е представена в *Структура на процесите*, където може да се проследи постъпково как тя се извършва, както и има отделен модул в декомпозицията, отговарящ за тази операция – *Записване на курсове* в *Операции на студенти*.

8. Студентите могат да генерират различни видове официални справки за студентския им статус: уверения, академични справки и т.н.

Функционалността е обособена в отделен модул *Генериране на справки* в *Студентски операции* в *Декомпозицията на модули*. Данните от справките се съхраняват в отделен модул *Справки* от *Конфигурацията на базата данни*. Проследяването на обмена на тази информация може да се направи в *Структура на потока на данните*.

9. Официалните справки са електронни, като трябва да са защитени от опит за фалшифициране. При желание, справките може да се разпечатват и на хартия, като хартиеното копие трябва да има механизъм за верифициране с електронния вариант на справката.

Логиката за верификация е представена в модула *Верификация на справки* в *Сигурност* от *Декомпозицията на модули*. Защитата за фалшифициране се представя в *Структурата на потока на данните*, където е представен правилния достъп до данните за коригиране, които имат само Административен, Учебен отдел и Преподаватели, а Студентите могат да достъпват данните единствено с цел преглеждане.

10. Системата да поддържа електронни студентски книжки, които са част от студентския профил. В тях, преподавателите внасят оценките на студентите по записаните от тях дисциплини, а студентите може да преглеждат своите книжки.

Това е реализирано чрез модула *Студентски книжки* в модула *Профили на студенти* от *Конфигурацията на базата данни*, като има и отделен модул в *Профили* в междинния слой отговарящ за логиката за изпълнение на тази функционалност. Процесът по нанасяне на оценка може да се проследи в *Структурата на процесите*, а обменът на данните от студентските книжки и тяхното достъпване в *Структурата на потока на данните.*

11. Системата да поддържа механизъм за публикуване на публични събития (еднократни курсове, състезания, събирания на групи по интереси и т.н.), които да може да се създават от всички потребители.

За тази функционалност отговаря модул *Управление на публични събития* в *Декомпозицията на модули*, а данните свързани с тях са представени в *Структура на потока на данните* и те се съхраняват в отделен модул в *Конфигурацията на базата данни.*

12. Системата да поддържа възможност за обмяна на лични съобщения между потребителите.

За тази функционалност отговаря модул *Комуникация между потребителите* в *Декомпозицията на модули*, а данните свързани с нея са представени в *Структура на потока на данните* и те се съхраняват в отделен модул в *Конфигурацията на базата данни.*

13. Потребителите от студентския съвет, както и преподавателите могат да създават заявки за различни искания, които се преглеждат и одобряват от потребителите в административния отдел.

За логиката на тази функционалност отговаря модул *Създаване на заявки* в *Операции на Студентски съвет* и *Преподаватели* и модул *Управление на предложения и искания* в *Административни операции*. Достъпа и процеса на пренасяне на този тип данни е показан в *Структура на поток на данните*.

14. Системата да поддържа защита на всички лични и финансови данни от неоторизиран достъп.

За защита и сигурността на данните в системата отговаря модул *Защита* в сървърната част от *Декомпозицията на модули*. Подмодулите за *Автентификация* и *Оторизация на потребители* съдържат логиката за тези процеси, които се използват с цел контролиране на достъпа да данните. За тяхната защита отговаря и модул *Криптиране на данни*, който осигурява съхранението на данните в криптиран вид.
По-добра защита за данните се осигурява и от разделянето на базата данни в отделен модул, който няма пряка връзка с презентационния слой, комуникиращ директно с потребителите.

15. Системата да предоставя API (публичен интерфейс) за достъп до генерираните официални справки и публични събития.

Интерфейсът за публичните събития е включен като част от модулите в *Презентационния слой* от *Декомпозицията на модули*, тъй като всеки потребител има достъп до публичните събития, създадени в системата.

Интерфейсът за генерираните официални справки е част от интерфейса на студентите, представен в модула *Студенти* от *Презентационния слой* в *Декомпозицията на модули*.

16. Системата трябва да е достъпна 24/7, като изключение за поддръжка и планирано обновяване се допуска само по време на официални празници.

В *Структурата на потока на данните* правилното им разпределение, както и използването от страна на потребителите само на необходимите за техните дейности данни ще подобри работата на системата с оглед на нейното бързодействие и надеждност, като не протича излишно количество данни при изпълнението на определен процес в системата и се поддържа тяхната коректност, като се предоставя достъп до тях от потребителите, които имат нужните права. Друга стъпка в осигуряването на висока надеждност на системата са модулите *Оторизация* и *Автентификация на потребителите* в *Декомпозицията на модулите*, които осигуряват достъп до данните само на потребителите, които имат необходимите правомощия.

17. Системата трябва да прави връзка със следните външни системи:
a. Държавни публични регистри за текущи студенти, към които периодично (напр. 2 пъти годишно) се изпраща информация за статуса на студентите. Изпращаната информация се контролира от потребителите от учебен и административен отдел.

b. Система за контрол на национална агенция за приходите и данъчната администрация.

c. Система за управление на учебното съдържание (напр. Moodle, но може и да е друга система, която се употребява в конкретния университет)

d. Списъкът с външни системи, с които се прави връзка може да се увеличи в процеса на използване на системата.

Отговорност за връзката с външните системи има модула *Управление на външни системи* от *Декомпозицията на модули*. Изискването за възможност за добавяне на нови външни системи е реализирано именно чрез обособяването на този модул, който съдържа логиката за тази функционалност. По този начин е улеснен самия процес на добавяне на нова система. За правилната интеграция спомага и *Структурата на потока на данните* и *Структура на процесите*, от която могат да се проследят данните в системата и как отделните модули са свързани с тях и комуникират помежду си.

18. Системата да може да издържа на пикови натоварвания (например увеличаване на потребителските заявки по време на кампания за записване на изборни дисциплини, вписване на оценки по време на сесия и т.н.), като може да обработва едновременно 1000 заявки в секунда.

Използването на трислойна архитектура за *Декомпозиция на модулите* и разпределянето на функционалностите на системата в отделни независими слоеве спомага за постигането да добра производителност, тъй като по този начин coupling-а е сведен до по-ниско ниво, с което се подобрява качеството на системата, а от там и на нейната производителност.
Друго решение, засягащо производителността и надеждността на системата, е разделянето на данните в две отделни бази – за профилите на потребители и останалите данни на системата. По този начин в критични моменти, при опити за атаки към системата и евентуалното сриване на функционалността за вход и достъпване до данните в базата за профилите на потребителите, то останалата част ще продължи цялостната си работа, като останалите модули няма да бъдат засегнати.
Правилното разпределение на ресурсите и данните в системата и контролирането на достъпа до тях от страна на потребителите в *Структура на потока на данните* подобрява ефективността на системата, а оттам и нейната производителност и повишаването на нейното качество.