Проект по

„Анализ и проектиране на информационни системи“



## **Тема: „Информационна система за управление на зали”**

спец. „Информационни системи“ - 3-курс

Ръководител: проф. д-р Калинка Калоянова

Участници:

Валентин Георгиев ФН: 71563

Георги Димов ФН: 71572

Александър Танков ФН: 71492

Ивелин Тодоров ФН: 71547

Богомил Богомилов ФН: 71591

Иван Атанасов ФН: 71560

Иво Райков ФН: 71561

### Съдържание:

Заглавие и Участници……….……………………..……….…….…...1

Съдържание……………………………………………...…………......2

Методи за определяне на изискванията и разпределение на

хората от екипа…….…………………………………………………...3

Записки на работата на екипа по определяне на изискванията..3

Кратко текстово описание на проекта………….……………………4

Начална визия на продукта: функционалност, ограничения,

елементи на потребителски интерфейс…………………………….4

Индивидуални модели на работа…………………………………….10

Use case модел………………………………………………………….17

Use case шаблон……………………………………………………..…18

Актьори и роли………………………………………………………….. Начален списък на потребителските случаи………………………..

Шаблон за пълно описание на потребителските случаи ……..

Пълно описание на 10% от потребителските случаи ………...

Елементи на потребителски интефейс, асоцииран с

потребителските случаи

Начално описание на нефункционалните изисквания по модела FURPS

Речник

Времето за работа на всеки студент по отделните

задачи…………………………………………………………....……….15

## 

## **Методи за определяне на изискванията и разпределение на хората от екипа**

За да определим какво точно иска от нас клиентът ни, ще приложим традиционни методи за определяне на изискванията. Те ще включват добре познатото интервю, което ще представлява среща с клиента очи в очи и дискутиране какви са неговите изисквания, да споделим какви са нашите виждания и какво можем да му предложим. Можем да разглеждаме вече съществуващи системи и да дискутиране с клиента дали му допада нещо конкретно и какво би искал да се заимства от тях като допълнителна функционалност към вече предложената от нас. Интервюто ще е формално, като ще представлява смесица от отворено и затворено интервю – ще има част с отговори на предварително дефинирани въпроси, както и част с неформално дискутиране на подробности, свързани със системата. За специфичния случай ще е добре да се допитаме и до лекторите, защото и те като хора които ще използват системата , може да има изисквания, които не трябва да се пренебрегват.

## 

## 

## 

## 

## **Записки на работата на екипа по определяне на изискванията**

Ще представим записките на екипа под формата на съвкупност от наблюденията ни по време на определяне изискванията за изграждане на системата. Всеки един от нас разгледа различна система, аналогична на нашата, с цел извличане на характеристики, които бихме могли да интегрираме директно или чрез малки корекции и подобрения. Забелязахме как повечето от разгледаните примери в основата си лежат на едни и същи принципи, като разликите бяха или козметични, или леки технически модификации на вече съществуващ такъв. Предвид опита на всеки един от нас, решихме да заложим на добре изпитан модел, в който се акцентира върху гъвкавост и надеждност. Първоначалното ни виждане за създаването на системата претърпя известна промяна след разглеждането на вече съществуващи подобни имплементации, което донякъде ни помогна и да дефинираме изискванията.

В последствие решихме да се консултираме с отговорното лице на ФМИ и да разберем как точно седят нещата в конкретната организация. След задаване на поредица от въпроси успяхме да добием подробна представа за това, как работи системата в университета.При изготвянето на графика лекторите не участват пряко в избора на ден, час или зала. Само когато някой лектор по- взискателен, тогава се комуникира с него и се взима и неговото мнение под въпрос.

## **Кратко текстово описание на проекта**

Проекта „Информационна система за управление на зали” е свързан с изграждането на платформа, която има за цел да улесни процеса около разпределянето на зали в различни учреждения като университети, училища и др. Тя ще предоставя удобен интерфейс на клиента, чрез който той лесно ще може да види даден календар (на стая,специалност, лектор и др.), да добавя различни изисквания при разпределянето на залите или да ги запазва ръчно.

Потребители на системата ще бъдат преподавателите и хората, занимаващи се с разпределянето на залите преди сесия или семестър.

## **Начална визия на продукта: функционалност, ограничения, елементи на потребителски интерфейс**

### Функционалности:

* Запитване за запазване на зала от лектор
* Вход в системата
* Редактиране на резервация от лектор.
* Изтриване на резервация от лектор.
* Промяна на лични данни
* Промяна на забравена парола.
* Изтегляне на информация
* Добавяне на лектор
* Изтриване на лектор
* Добавяне на нова стая.
* Добавяне на нови специалности.
* Преглед на детайлите на системата.
* Редактиране на резервация от администратор
* Запазване на зала от администратор
* Промяна на парола
* Редактиране на лектор от администратора
* Премахване на стая
* Редактиране на стая
* Премахване на специалност
* Редактиране на специалност
* Интеграция със система, ползваща се за генериране на учебния график

### Системата трябва да поддържа следните нефункционални изисквания:

* Да е достъпна през уеб интерфейс
* Да поддържа около 100-150 посещения на ден
* Да поддържа 1 администраторски акаунт
* Да поддържа 1000 лекторски акаунта
* Да осигурява зареждане на календара до 1-1.5 секунди
* Да поддържа паролите в криптиран вид
* Да може да е в експлоатация 10 години
* Лекторските сесии в системата изтичат автоматично при 1 час неактивност на потребителя
* Да има автоматичен онлайн бекъп не по-рядко от веднъж на 24 часа

В основата си, нашата система е специфично насочена към университета и по-конкретно, към ФМИ. Тя ще предоставя точен график на задължителните предмети от текущия семестър, както и на избираемите предмети. Преди сесия ще могат да бъдат видими и графиците за изпитите, къде ще се състоят и от колко часа.

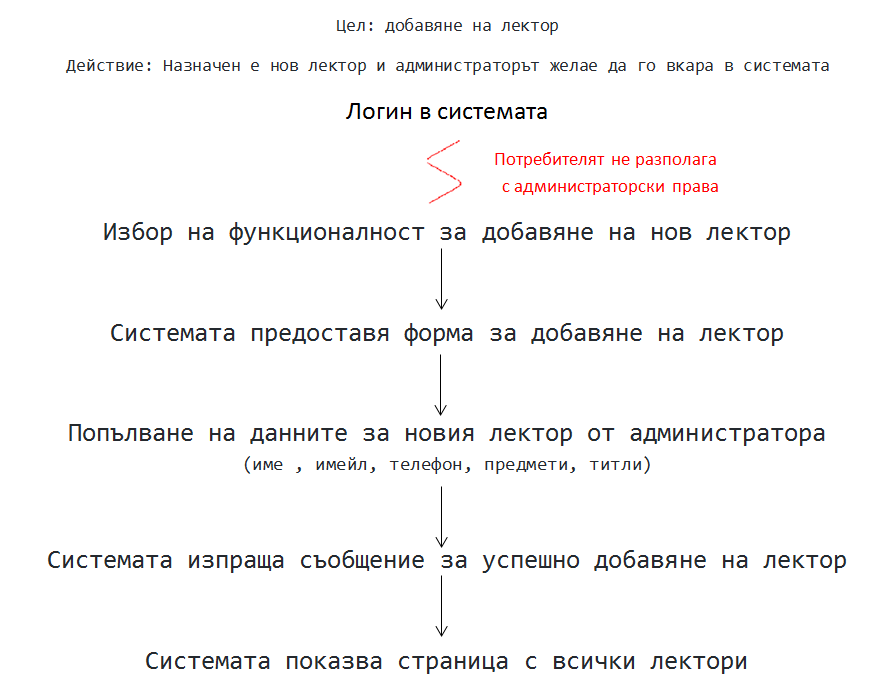
Съставянето на графиците за семестриалната година ще бъде извършена в няколко етапа:

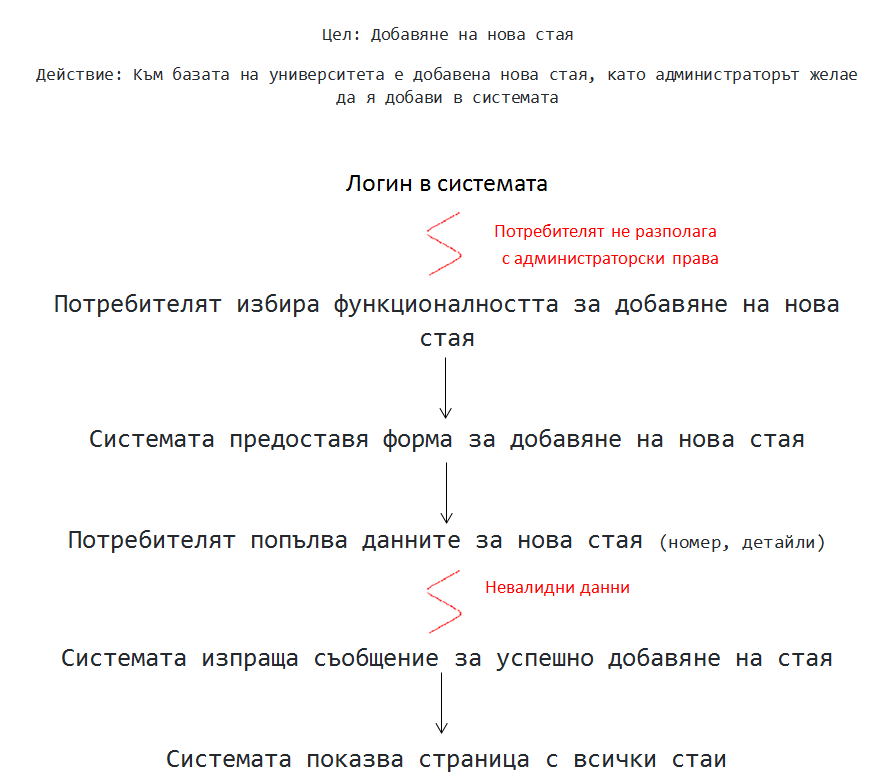
1. В 10 дневен срок лекторите, които имат предпочетания относно деня, датата и мястото на тяхното занятие ще имат възможност да попълнят форма.
2. След 10 дневния срок администратора, комуникира с лекторите (ако е нужно).
3. За останалите занятия ,където не е имало активност на лектор, на случаен принцип се генерират събития, които се въвеждат в календара.
4. След стъпка 1, 2, 3, календарите на лекторите, стаите и специалностите вече са изготвени за текущия семестър.
5. За избираемите курсове, допълнителните упражнения и консултации ще се разчита на всеки лектор да е попълнил формата за график, след което администратора да ги е одобрил.
6. Преди сесия, всеки лектор трябва в 10 дневен срок да въведе своите предпочетания, които са съобразени със студентите.
7. След 10 дневният срок администратора допълнително комуникира с лекторите( ако е нужно ).
8. Графика е изготвен спрямо заявленията на лекторите.

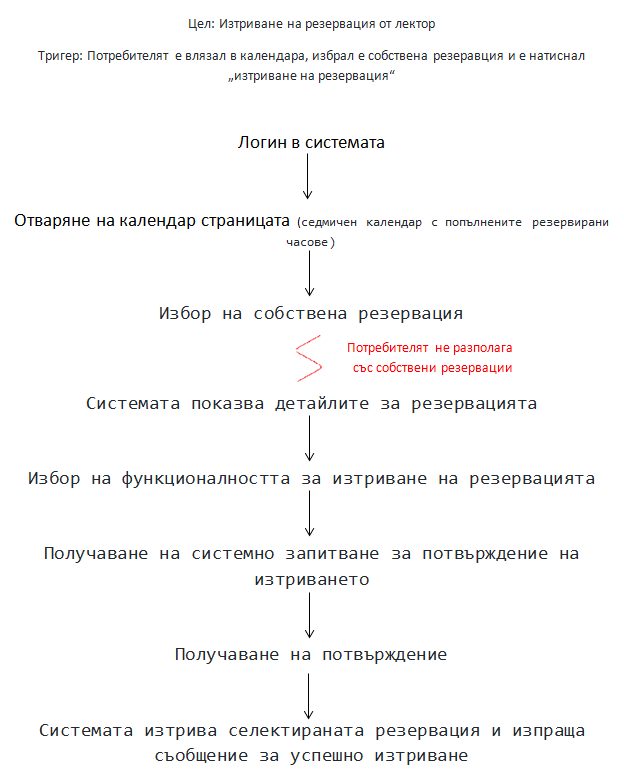
Учебния график ще представлява съвкупност от записи, които ще съдържат в себе си следните атрибути: Title, Room, Date, Duration, Start/end time, specialty. Ще се предлага функционалност, чрез която да се преглеждат календари за специалността, на избран лектора или на определена стая. Трите календара ще се съобразяват един с друг, така че да не се получава застъпване на заниманията в календарите. Поради факта че консултациите се извършват веднъж, ще има фунционалност, чрез която единично да се добавя събитие в календара. Дадено събитие ще може да се конфигурира, така че да се повтаря през определен период от време(сесия) или да съществува за избраната дата.

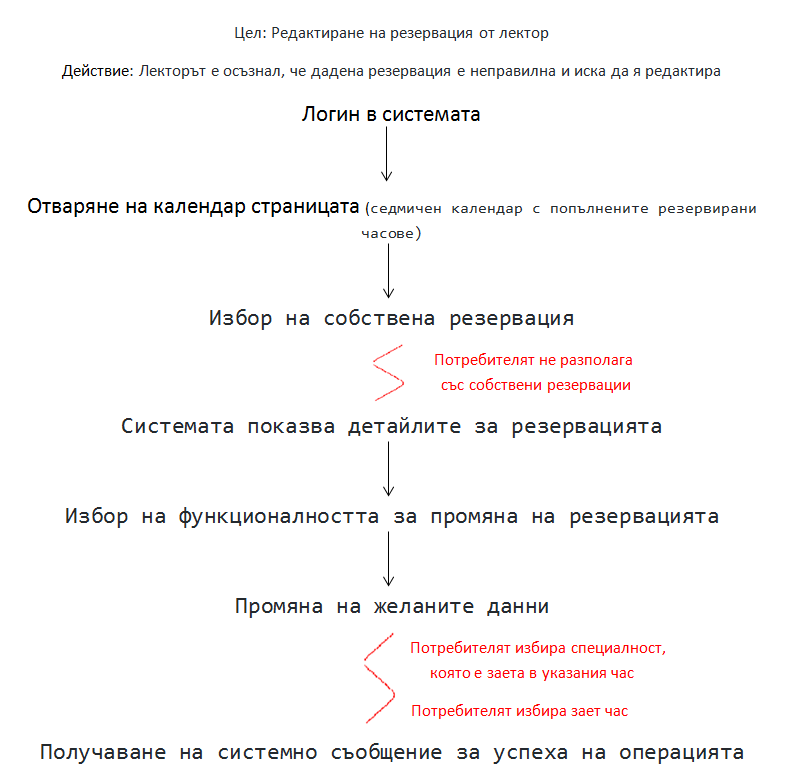
При създаването на стая администратора задава номер и тип на стаята - дали е за лекции или упражнения( с компютърна база). Друга доста полезна функционалност ще бъдe възможността да се добавят ресурси към съответна стая. Администратора ще може да конкретизира дали в стаята има видео проектор, бяла дъска или екран за презентиране. Всяка стая трябва да притежава атрибут Capacity, за да се избегне случай, в който стаята е твърде малка за специалността. Всяка стая има собствен календар, който може да бъде изтеглен, като отделен ресурс.

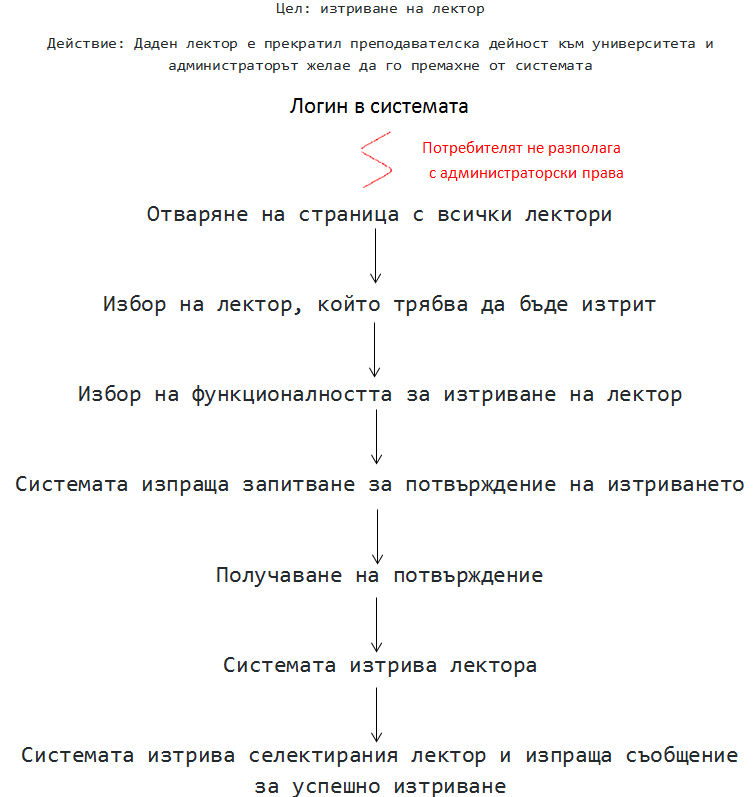
## **Индивидуални модели на работа**

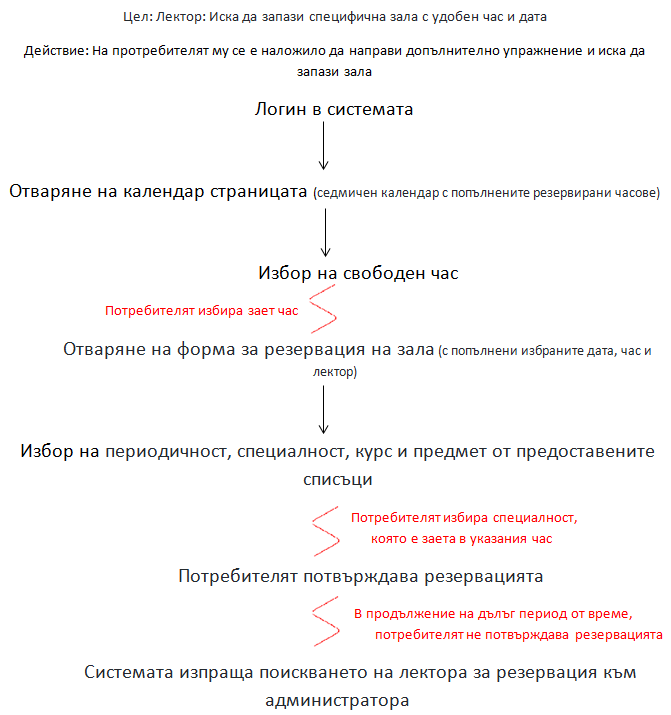


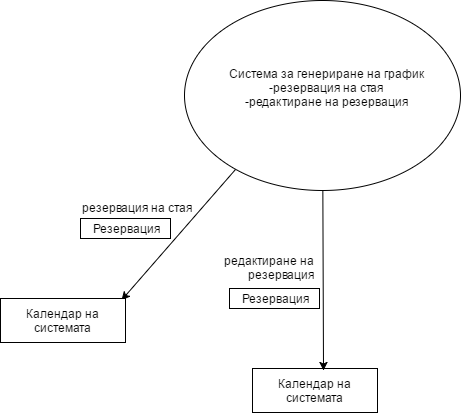


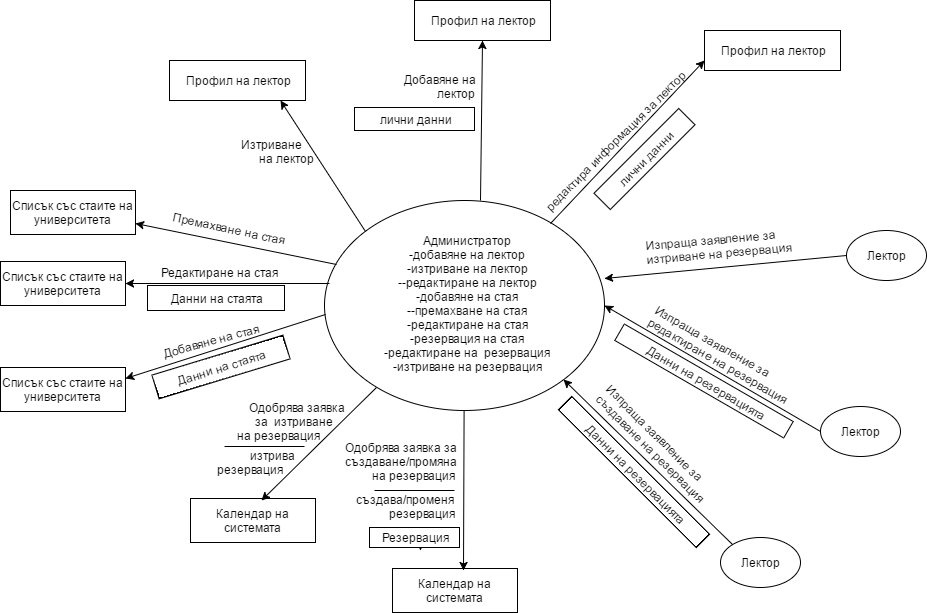


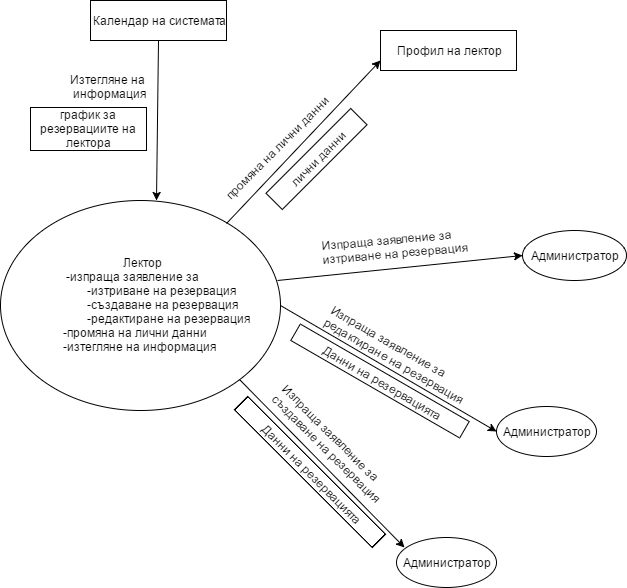










****

**Актьори и роли**

Системата трябва да поддържа следните актьори:

* Лектор:
  + Който трябва да може:
    - Да се вписва в системата
    - Да проверява за празна стая
    - Да прави запитване за резервация на стая
    - Да редактира и изтрива собствена резервация
    - Да може да си изтегля някаква информация (например календари) в pdf формат
    - Да променя личните си данни (email, телефон, предмети)
    - Да промяна паролата си
* Администратор:
  + Който трябва да може:
    - Да има всичките права на лектора
    - Да създава, редактира и изтрива акаунти на лектори
    - Да разглежда личните данни на лекторите, въведени в системата
    - Да добавя, редактира, изтрива нови стаи
    - Да добавя, редактира и изтрива нови специалности
    - Да вижда списък с лекторите, стаите и специалностите в системата
    - Да изпраща имейл с покана до лекторите на факултета

Ролите, като част от домейн модела трябва да са следните:

* Стая – всяка една стая в университета, която може да се ползва, ще бъде добавена в системата от администратора. Така лекторите ще могат лесно да запазват различни стаи в зависимост от техните изисквания. Всяка стая ще съдържа следната информация в себе си:
  + Тип (с мултимедия, с компютри и т.н)
  + Номер
  + Капацитет
  + Заетост – Това ще представлява собствен календар на стаята, в който може да се провери тази стая кога е свободна и кога не. Ако е запазена, ще има информация кой я е запазил, за коя специалност и какъв точно е предметът.
* Специалност:
  + Година
  + Брой хора (като може лесно да се вземе от броя на хората в групите)
  + Групи – Представлява различните групи от хора в специалността като всяка една група ще съдържа следната информация:
    - Номер
    - Брой хора
    - Учебен план – Той описва какви точно ще са задължителните
* Предмети
  + Седмична заетост (колко на брой лекции и упражнения ще имат за една седмица по даденият предмет)
  + Лектор – Кой ще я води
  + Семестър(зимен/летен) – в кой курс и семестър ще се изучава
  + Курс и специалност (за задължителните предмети)
* Резервация
  + Име – Когато се избира за даден предмет то ще бъде автоматично генериран и ще съдържа името на специалността, името на предмета и името на лектора. Ако не се избира за даден предмет (например за консултация) авторът на резервацията попълва името.
  + Кога – в кой ден и кой час
  + Къде – в коя стая
  + Допълнителна информация

## **Списък на потребителските случаи**

## Екипно решихме да приоритизираме всички пълни описания на потребителските случаи по това колко често ще се използват в информационната система за разпределение на зали

## **Потребителски случаи с приоритет А.**

А-1. Поискване на резервация на зала от лектор:

Всеки лектор в системата има право да поиска да запази зала. Когато той избере час, периодичност и зала, трябва да попълни заглавие на заниманието, което ще се провежда, курс и специалност. В резултат системата изпраща до администратора поискването на лектора и той преценява дали да запази тази зала за него или не.

А-2. Вход в системата:

Всеки потребител на информационната система трябва да може да влезе в нея. След успешно влизане в системата, в зависимост от ролята, потребителят има различни възможности и права.

## **Потребителски случаи с приоритет B.**

Б-1. Редактиране на резервация от лектор:

Лекторът, който е запазил зала, има възможност да променя детайлите по тази резервация. Това се прави с цел ако трябва да се променят часа или залата за занятието. В резултат на промяната има проверка дали залата е свободна за указаната дата. Ако е свободна се променя резервацията, ако не е се съобщава на потребителя.

Б-2. Изтриване на резервация от лектор:

Лекторът, който има запазена зала, има възможност да премахне резервацията и да освободи залата. Целта е, ако лекторът не може да присъства на заниманието да го отмени и освободи залата.

Б-3. Изтегляне на информация:

Всеки потребител има възможност да изтегли информация за графика си по месец/седмица/специалност/курс в pdf формат. След успешно изтегляне, лектора може да разпечата графика си или да го изпрати на студенти.

Б-4. Добавяне на лектор:

Администраторът има права и възможност да добавя нови лектори към факултета. Ако нов колега бъде назначен на работа, да може администраторът лесно и удобно да му създаде акаунт с който лектора да има достъп до системата.

Б-5. Редактиране на резервация от администратор:

Лекторът, който е запазил зала, има възможност да променя детайлите по тази резервация. Това се прави с цел ако трябва да се променят часа, периодичността или залата за занятието. В резултат на промяната има проверка дали залата е свободна за указаната дата. Ако е свободна се променя резервацията, ако не е се съобщава на потребителя.

**Потребителски случаи с приоритет C.**

C-1. Запазване на зала от администратор:

Всеки администратор в системата има право да запази зала. Когато той избере час и зала, трябва да избере лектора, който ще я води. Също така трябва да попълни заглавие на заниманието, което ще се провежда, курс и специалност. В резултат системата запазва залата и блокира възможноста да се правят други резервации за резервирания час и място от други лектори.

C-2. Промяна на лични данни:

Всеки лектор има възможност да промени личните си данни. Да добави титла, редактира парола, да смени имейл, телефон или да добави предметите, които провежда. В резултат на това, всеки друг потребител ще види новите данни на лектора.

C-3. Промяна на забравена парола:

Ако потребителя е забравил паролата си, има възможност да я поднови с нова на ниво вход в системата. Новата парола се генерира динамично от системата и се изпраща на имейла на лектора. След успешно влизане в системата с новата парола, лекторът има възможност да я замени с друга парола.

C-4. Изтриване на лектор:

Администраторът има възможност да премахне лектор от списъка с действащи лектори. Това се прави с цел, ако даден лектор напусне или се пенсионира.

C-5 . Добавяне на нова стая:

Администраторът може да добавя нова стая към списъка със стаи. Целта е, когато има разширяване на факултета да може да се добавят лесно и удобно нови стаи с техните детайли.

C-6. Добавяне на нови специалности:

Администраторът има възможност да добавя нови специалности. В резултат на това списъка със специалности, които лекторите виждат като запазват стаи ще е напълно актуален.

C-7. Промяна на парола:

Ако потребителят иска да смени паролата си, има възможност да го направи след като се е вписал в системата. От полето за смяна на личните данни избира да смени паролата си. Първо въвежда старата парола а след това новата и потвърждава с натискането на бутон за потвърждаване.

C-8. Редактиране на лектор от администратора:

Администраторът има възможност да променя данните на лектори във факултета. Ако случайно е допусната някаква грешка да не се налага да пресъздава акаунта от начало.

C-9. Премахване на стая:

Администраторът има възможност да премахва дадена стая. В случай, че някаква стая се ремонтира или вече не съществува да бъде премахната от системата и да не може да се запазват занятия за нея.

C-10. Редактиране на стая:

Администраторът има възможност да редактира данните на дадена стая. В случай, че някаква стая вече има различни характеристики (има компютри или мултимедия за пример) да може да бъде променена в системата.

C-11. Премахване на специалност:

Администраторът има възможност да премахва специалности. В случай, че някаква специалност не се предлага вече в университета да бъде премахната от системата и да не се генерират програми за нея.

C-12. Редактиране на специалност:

Администраторът има възможност да редактира данните на специалностите. В случай на промяна на някакви данни да не се налага да се пресъздава специалността.

## **Use case модел**

## Mодела на потребителските случаи показва системните изисквания в контекста на потребителския случай. Като актьори на модела са представени лектора и администратора, които могат да попаднат в следните потребителски случаи.

## 

## **Use case шаблон**

потребителски случай <номер на UC>: <име на UC>

ниво: user-goal или subfunction

актьор: ...

заинтересовани страни:

- един: какво е важно за него

- друг: ...

действие:

предпоставки: (какво трябва да е изпълнено преди началото и си заслужава да го кажем)

- едно

- друго

последствия: (какво трябва да е вярно след успешен завършек и си заслужава да го кажем)

- едно

- друго

главен успешен сценарий: (типичния, безусловен "хепи" път на успешен сценарий)

1. действие

2. действие

...

разширения: (алтернативни сценарии (успешни/не))

2a. грешен логин

1. нещо

2. нещо

5а. друга алтернатива

1. нещо

2. нещо

специални изисквания:

- един

- друг (нефункционално изискване)

честота на използване: (оказва влияние на това кога ще задълбаем в проучването, реализацията, тестването и т.н.)

много

по-малко

много по - малко

други:

забележки, въпрос

**потребителски случай A-1: Поискване на резервация на зала от лектор**

ниво: Потребителска цел

актьор: Лектор

заинтересовани страни:

Лектор: Иска да запази специфична зала с удобен час и дата

действие:

На потребителят му се е наложило да направи допълнително упражнение и иска да запази зала

предпоставки:

Лекторът трябва да е влязъл в системата като лектор

последствия:

Инициаторът е поискал да запазил успешно залата и се чака отговор от администратора.

главен успешен сценарий:

01. Потребителят отива на календар страницата

02. Системата предоставя седмичен календар с попълнени резервираните часове

03. Потребителят избира свободен час.

04. Системата предоставя форма за резервация на зала с попълнени избраните дата, час и лектор.

05. Потребителят избира периодичност, специалност, курс и предмет от предоставените списъци.

06. Потребителят потвърждава резервацията.

07. Системата изпраща поискването на лектора за резервация към администратора.

разширения:

03a. Потребителя избира зает час

01. Системата информира потребителя, че няма как да запази вече зает час.

02. Продължаваме със стъпка 03 от главния сценарий.

06а. Потребителя избира специалност, която е заета в указания час.

01. Системата информира потребителя, че специалноста е заета.

02. Продължаваме от стъпка 03 от главния сценарий.

07а. В продължение на дълъг период от време потребителят не потвърждава резервацията.

01. Системата премахва резервацията на стаята за този час

02. Продължаваме със стъпка 02 от главния сценарий.

специални изисквания:

- системата запазва залата за по-малко от 0.8 секунди.

честота на използване:

- много

други: N/A

# **потребителски случай A-2: Вход в системата**

# ниво: Потребителска цел

# актьор: лектор, администратор

# заинтересовани страни:

# - Лектор: Иска да влезе в системата

# - Администратор: Иска да влезе в системата

# действие:

# Потребителят е достъпил системата като гост и иска да може да я използва като лектор или администратор

# предпоставки:

# Потребителят да е заредил информационната система и тя е предоставила форма за вход в системата

# последствия:

# Потребителят е попълнил правилно формата и получава достъп до системата.

# главен успешен сценарий:

# 01. Потребителят попълва потребителско име и парола

# 02. Потребителят избира функционалност за вход

# 03. Системата проверява валидността на данните

# 04. Системата предоставя достъп до функционалностите в зависимост от ролята

# 

# разширения:

# 

# 03a. Потребителя попълва грешни данни

# 1. Системата показва съобщение за грешни входни данни.

# 2. Продължаваме със стъпка 01 от главния сценарий.

# специални изисквания:

# - системата проверява и дава достъп на потребителя за по-малко от 1.4 секунди.

# честота на използване:

# - много

# други: N/A

# 

**потребителски случай B-1: Редактиране на резервация от лектор**

ниво: Потребителска цел

актьор: лектор

заинтересовани страни:

- Лектор: Иска да промени данните на резервация, която сам е направил

действие:

Лекторът е осъзнал, че дадена резервация е грешна и иска да я редактира

предпоставки:

Лекторът трябва да е влязъл в системата и да има поне една резервация

последствия:

Лекторът е променил успешно данните на резервацията.

главен успешен сценарий:

01. Потребителят отива на календар страницата

02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

03. Потребителят избира собствена резервация

04. Системата показва детайлите на резервацията

05. Потребителят избира функционалноста за промяна на резервацията

06. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията с изключение на датата и часа.

07. Потребителят променя данните, които желае

08. Системата проверява дали указаните данни са правилни

09. Системата показва съобщение за успешно променена резервация

разширения:

08а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 07 от главният сценарий

специални изисквания:

- системата променя резервацията за по-малко от 1 секунда.

честота на използване:

- по-малко

други:

N/A

**потребителски случай B-2: Изтриване на резервация от лектор**

ниво: Потребителска цел

актьор: лектор

заинтересовани страни:

Лектор: Иска да изтрие резервация, която сам е направил

действие:

Лекторът е преценил, че няма да може да се проведе дадено занятие и затова е решил да изтрие резервацията за това занятие

предпоставки:

Лекторът трябва да е влязъл в системата и да има поне една резервация

последствия:

Лекторът е изтрил успешно резервацията и е информиран за това.

главен успешен сценарий:

01. Потребителят отива на календар страницата

02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

03. Потребителят избира собствена резервация

04. Системата показва детайлите на резервацията.

05. Потребителят избира резервацията, която иска да изтрие.

06. Системата пита за потвърждение за изтриването на резервацията.

07. Потребителят потвърждава изтриването на резервацията.

08. Системата изтрива резервацията

09. Системата изпраща съобщение за успешно изтриване.

разширения:

07а. Потребителя отказва изтриване на резервация

- Продължаваме със стъпка 04 от главния сценарий

специални изисквания:

- системата изтрива резервацията за по-малко от 0.7 секунда.

честота на използване:

- по-малко

други:

N/A

**потребителски случай B-3: Изтегляне на информация**

ниво: Потребителска цел

актьор: лектор, администратор

заинтересовани страни:

- Лектор: Може да изтегли графика си със всички запазени от него зали за определен период от време.

- Администратор: Може да изтегли графика на всеки лектор, зала, специалност и курс за определен период от време.

действие:

Лектор/Администратор иска да си свали график на стая,специалност или собствен за определен период от време

предпоставки:

Потребителят е влязъл в системата като лектор/администратор и е влязъл в профила си.

последствия:

Изтегля график успешно за избрания период от време.

главен успешен сценарий:

01. Потребителя отива на изтегляне на график.

02. Системата предоставя опции за изтегляне на график (администратор: курс/специалност/стая/лектор и период от време: месец, седмица) (лектор: период от време: месец/седмица).

03. Потребителя попълва данните, за които иска да изтегли информация

04. Системата изтегля графика

разширения:

N/A

специални изисквания:

Системата изтегля графика за около 10 секунди

честота на използване:

- по-малко

други:

N/A

**потребителски случай B-4: Добавяне на лектор**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Иска да добави нов лектор

действие:

Назначили са нов лектор и администратора иска да го добави в системата, за да може да я ползва

предпоставки:

Потребителят е влязъл в системата като администратор

последствия:

Добавен е нов лектор успешно и е показана страницата с всички лектори

главен успешен сценарий:

01. Потребителят отива на добавяне на нов лектор

02. Системата предоставя форма за добавяне на лектор.

03. Потребителят попълва данните за новия лектор (име, имейл, телефон, предмети, титли)

04. Системата валидира попълнените данни

05. Системата изпраща съобщение за успешно добавяне на лектор

06. Системата показва страница с всички лектори.

разширения:

04а Валидацията на данните е неуспешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 02 от главният сценарий

специални изисквания:

Системата добавя нов лектор за около 1 секунда

честота на използване:

- по-малко

други:

N/A

**потребителски случай B-5: Редактиране на резервация от администратор**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Иска да промени данните на резервация

действие:

Администратора е пожелал да промени дадена резервация, защото нещо в нея не е коректно

предпоставки:

Потребителят трябва да е влязъл в системата като администратор

последствия:

Потребителят е променил успешно данните на резервацията.

главен успешен сценарий:

01. Потребителят отива на календар страницата на специалност/стая

02. Системата предоставя седмичен календар с попълнени резервираните часове

03. Потребителят избира резервация

04. Системата показва детайлите на резервацията.

05. Потребителят избира функционалноста за промяна на резервацията.

06. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

07. Потребителят променя данните, които иска да промени.

08. Системата проверява дали указаните данни са правилни (проверява дали часа и залата са сменени с други и дали са свободни и дали лекторът е свободен в този час )

09. Системата информира лектора за промянта

09. Системата показва съобщение за успешна промяна

разширения:

08а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 07 от главният сценарий

специални изисквания:

- системата променя резервацията за по-малко от 1 секунда.

честота на използване:

- по-малко

други:

N/A

**потребителски случай C-1: Запазване на зала от администратор**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Иска да запази специфична зала с удобен час и дата и лектор

действие:

Администраторът му се е наложило да запази зала за някакво занятие

предпоставки:

Потребителят трябва да е влязъл в системата като администратор

последствия:

Администраторът е запазил успешно залата и тя е резервирана за дадения час.

главен успешен сценарий:

01. Потребителят отива на календар страницата за дадения лектор

02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

03. Потребителят избира свободен час.

04. Системата предоставя форма за резервация на зала с попълнени избраните дата и час.

05. Потребителят избира зала, лектор, периодичност, специалност, курс и предмет от предоставените списъци.

06. Системата проверява данните

07. Системата запазва залата и показва на другите лектори, че часът и залата са запазени.

разширения:

03a. Потребителя избира зает час

01. Потребителя няма възможност да запази вече зает час.

02. Продължаваме със стъпка 02 от главният сценарий

06а. Потребителя избира специалност, която е заета в указания час.

01. Системата изпраща съобщение на потребителя, че специалноста е заета.

02. Продължаваме със стъпка 02 от главният сценарий

специални изисквания:

- системата запазва залата за по-малко от 0.8 секунди.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-2: Промяна на лични данни**

ниво: Потребителска цел

актьор: лектор

заинтересовани страни:

- Лектор: Иска да промени личните си данни (титли, телефон, имейл или да добави предмети, които ръководи)

действие:

Лекторът е влязъл в системата и желае да смени личните си данни

предпоставки:

Потребителят трябва да е влязъл в системата като лектор

последствия:

Потребителят е променил успешно личните си данни.

главен успешен сценарий:

01. Потребителят отива на профил страницата

02. Системата предоставя личните данни, които има лектора

03. Потребителят променя данните, които желае.

04. Системата валидира данните

05. Системата показва съобщение за успешно променени данни.

разширения:

04a. Валидацията на данните е неуспешна

01. Системата показва съобщение за грешка свързано с невалидните данни

02. Системата предоставя личните данни, които има лектора

03. Системата сигнализира кои полета са попълнени грешно

04. Продължаваме със стъпка 03 от главния сценарий

специални изисквания:

- системата променя данните на лектора за по-малко от 1 секунда.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-3: Промяна на забравена парола**

ниво: Потребителска цел

актьор: гост

заинтересовани страни:

- Гост: Ако потребителя е забравил паролата си и иска да я възстанови.

действие:

Потребителят е забравил паролата си и иска да я смени.

предпоставки:

Потребителят е достъпил системата като гост

последствия:

Потребителят е получил успешно новата парола.

главен успешен сценарий:

01. Потребителят избира функционалността за забравена парола

02. Системата показва форма, която съдържа поле за въвеждане на имейл, на който ще бъде изпратена новата парола.

03. Потребителят въвежда своя имейл.

04. Системата проверява имейла дали съществува в базата от данни

05. Системата генерира новата парола

05. Системата изпраща новата парола

разширения:

04a. Имейла, който е въведен не е намерен в базата от данни

01. Системата показва съобщение за грешка

02. Продължаваме със стъпка 02 от главния сценарий.

специални изисквания:

- системата изпраща нова динамично генерирана парола на имейла на потребителя за по-малко от 5 секунда.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-4: Изтриване на лектор**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

Администратор: Може да изтрива лектор, ако бъде уволнен или пенсиониран.

действие:

Лектор е уволнен и администратора иска да го премахне от системата

предпоставки:

Потребителят е влязъл в системата като администратор

последствия:

Изтриване на лектор успешно от администратора

главен успешен сценарий:

01. Потребителят отива на страницата, на която са показани всички лектори

02. Системата показва всички лектори.

03. Потребителят избира лектор, когото иска да изтрие.

04. Потребителят избира функционалността за изтриване на лектор

05. Системата изпраща системно запитване за потвърждение на изтриването

06. Потребителят потвърждава изтриването

07. Системата изтрива лектора

08. Системата изпраща съобщение за успешно изтриване на лектор

разширения:

N/A

специални изисквания:

Системата изтрива лектор за около 0.5 секунди.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-5: Добавяне на нова стая**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Може да добавя нова стая, ако бъде присвоена/построена такава.

действие:

Направена е нова стая и администраторът иска да я добави в системата

предпоставки:

Потребителят е влязъл в системата като администратор

последствия:

Добавена е нова стая успешно и страницата със стаите е визуализирана

главен успешен сценарий:

01. Потребителят отива на добавяне на нова стая

02. Системата предоставя форма за добавяне на стая.

03. Потребителят попълва данните за нова стая (номер, детайли)

04. Системата изпраща съобщение за успешно добавяне на стая

05. Системата показва страница с всички стаи.

разширения:

N/A

специални изисквания:

Системата добавя нова стая за около 1 секунда

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-6: Добавяне на нови специалности**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Може да добавя нова специалност, ако бъде създадена такава.

действие:

Нова специалност е добавена в факултета и администратора иска да я добави в системата

предпоставки:

Потребителят е влязъл в системата като администратор

последствия:

Добавена е нова специалност успешно

главен успешен сценарий:

01. Потребителят отива на добавяне на нова специалност

02. Системата предоставя форма за добавяне на специалност.

03. Потребителят попълва данните за нова специалност (курсове, капацитет, предмети)

04. Системата валидира данните

05. Системата изпраща съобщение за успешно добавяне на специалност

06. Системата показва страница с всички специалности.

разширения:

04а Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на специалността

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 03 от главният сценарий

специални изисквания:

Системата добавя нова специалност за около 1 секунда

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-7: Промяна на паролa**

ниво: Потребителска цел

актьор: лектор, администратор

заинтересовани страни:

- Лектор: Иска да промени паролата си и да въведе нова.

- Администратор: Иска да промени паролата си и да въведе нова.

действие:

Лекторът/Администраторът желае да си смени паролата.

предпоставки:

Потребителят е влязъл в системата като лектор/администратор

последствия:

Потребителят е променил успешно паролата си.

главен успешен сценарий:

01. Потребителят отива на профил страницата

02. Системата предоставя личните данни, които има лектора

03. Потребителят избира функционалността за смяна на паролата

04. Системата предоставя полета за попълване на паролата

05. Потребителят попълва полетата. за стара и нова парола

06. Системата валидира новата парола

07. Системата показва съобщение за успешно променена парола.

разширения:

04a. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени полетата с паролите.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 05 от главният сценарий

специални изисквания:

- системата променя паролата на лектора/администратора за по-малко от 0.5 секунди.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-8: Редактиране на лектор от администратора**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администраторът: Иска да промени данните на лектор

действие:

Администраторът случайно обърква данните на даден лектор и иска да ги редактира

предпоставки:

Потребителят е влязъл в системата като администратор и да има поне една лектор в системата

последствия:

Администраторът е променил успешно данните на лектора.

главен успешен сценарий:

01. Потребителят отива на страницата, която показва всички лектори и натиска лектора, който иска да редактира

02. Системата предоставя данните на лектора

03. Потребителят избира функционалноста за промяна на данните на лектора.

04. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

05. Потребителят променя данните, които иска да промени.

06. Системата проверява дали указаните данни са правилни

07. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

разширения:

06а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 05 от главният сценарий

специални изисквания:

- системата променя данните на лектор за по-малко от 1 секунда.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-9: Премахване на стая**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Може да премахва стая, ако бъде разрушена или е в ремонт.

действие:

Дадена стая вече не е годна за употреба и администратора иска да я премахне

предпоставки:

Потребителят е влязъл в системата като администратор

последствия:

- Изтриване на стая успешно

главен успешен сценарий:

01. Потребителят отива на страницата, на която са показани всички стаи

02. Системата показва всички стаи.

03. Потребителят избира стая, която иска да изтрие.

04. Системата пита за потвърждение за изтриването на стаята.

05. Потребителят потвърждава изтриването на стаята.

06. Системата изтрива стаята.

07. Системата изпраща съобщение за успешно изтриване.

разширения:

05а. Потребителя отказва изтриване на стая

- Продължаваме със стъпка 04 от главния сценарий

специални изисквания:

Системата изтрива стая за около 0.5 секунди.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-10: Редактиране на стая**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Иска да промени данните на стая

действие:

Дадена стая е поправена (примерно са добавени са компютри или мултимедия) и администраторът иска да го добави в системата като редактира данните на стаята

предпоставки:

Потребителят трябва да е влязъл в системата като администратор

последствия:

Администраторът е променил успешно данните на стаята.

главен успешен сценарий:

01. Потребителят отива на страницата, където са показани всички стаи

02. Системата показва всички стаи

03. Потребителят избира стая

04. Системата показва детайлите на стаята.

05. Потребителят избира да промени стая.

06. Системата предоставя възможност на потребителя да промени всяка една от данните на стаята.

07. Потребителя променя данните, които иска да промени.

08. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

разширения:

N/A

специални изисквания:

- системата променя детайлите на стая за по-малко от 1 секунда.

честота на използване:

- много по-малко

други:

N/A

**потребителски случай C-11: Премахване на специалност**

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Желае да премахне специалност

действие:

Дадена специалност е премахната и администраторът иска да я премахне от системата

предпоставки:

- Потребителят е влязъл в системата като администратор

последствия:

- Изтриване на специалност успешно

главен успешен сценарий:

01. Потребителят отива на страницата, на която са показани всички специалности

02. Системата показва всички специалности.

03. Потребителят избира специалност, която иска да изтрие.

04. Системата пита за потвърждение за изтриването на специалността

05. Потребителят потвърждава изтриването на специалността

06. Системата изтрива специалността

07. Системата изпраща съобщение за успешно изтриване.

разширения:

05а. Потребителя отказва изтриване на специалност

- Продължаваме със стъпка 04 от главния сценарий

специални изисквания:

Системата изтрива специалност за около 0.5 секунди.

честота на използване:

- много по-малко

други:

N/A

потребителски случай C-12: Редактиране на специалност

ниво: Потребителска цел

актьор: администратор

заинтересовани страни:

- Администратор: Иска да промени данните на специалност

действие:

Дадена специалност е променена (примерно в предметите, които се изучават) в факултета и администраторът иска да я промени в системата

предпоставки:

Потребителят трябва да е влязъл в системата като администратор

последствия:

Инициаторът е променил успешно данните на специалността.

главен успешен сценарий:

01. Потребителят отива на страницата, където са показани всички специалности.

02. Системата показва всички специалности

03. Потребителят избира специалност

04. Системата показва детайлите на специалността.

05. Потребителят избира функционалноста за промяна на специалност.

06. Системата предоставя възможност на потребителя да промени всяка една от данните на специалността.

07. Потребителят променя данните, които иска да промени.

08. Системата проверява дали указаните данни са правилни (курсове, капацитет, предмети)

09. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

разширения:

N/A

специални изисквания:

- системата променя детайлите на специалност за по-малко от 1 секунда.

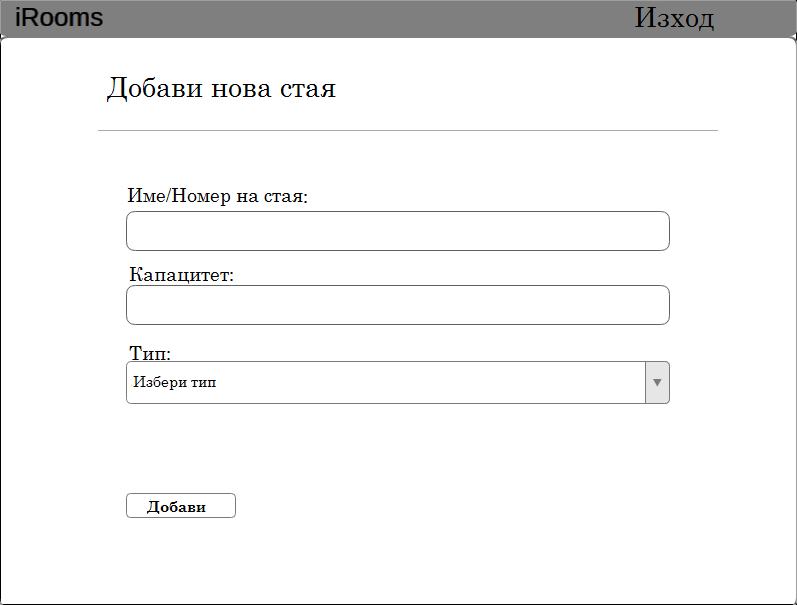
честота на използване:

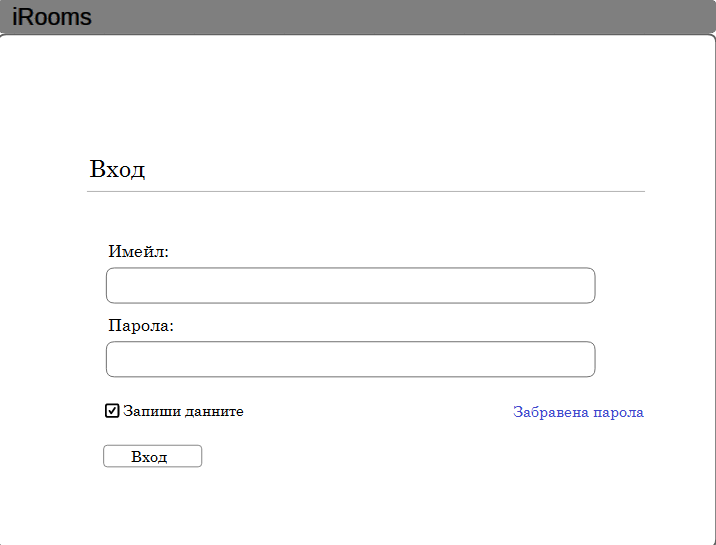
- много по-малко

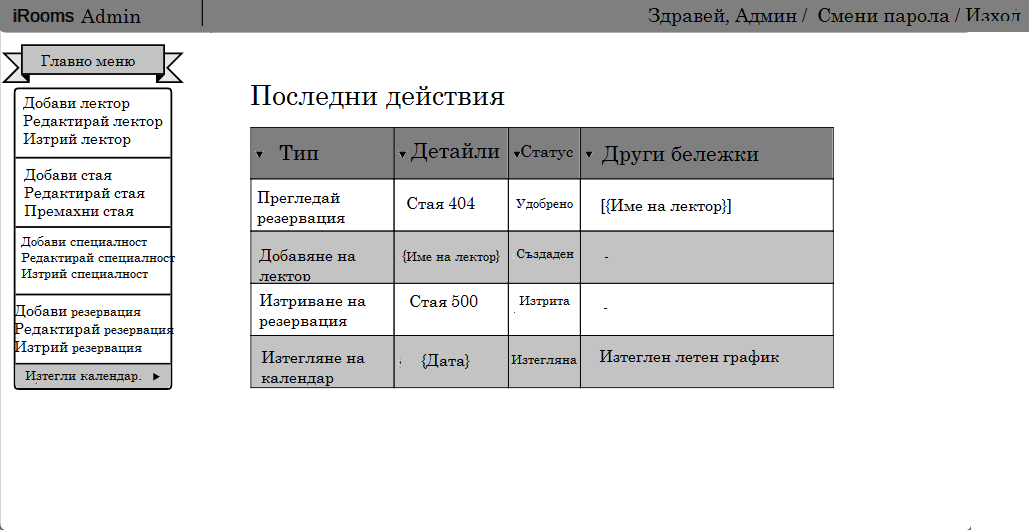
други:

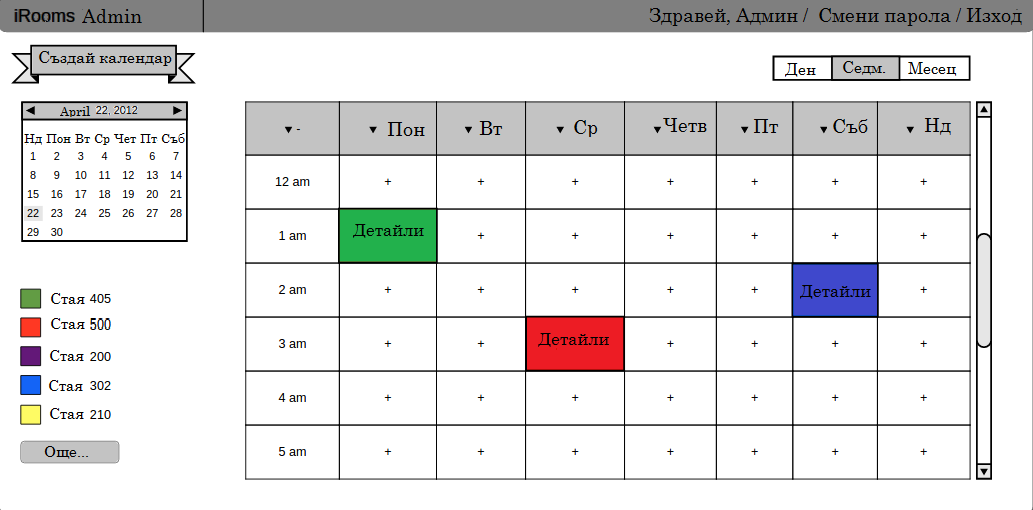
N/A

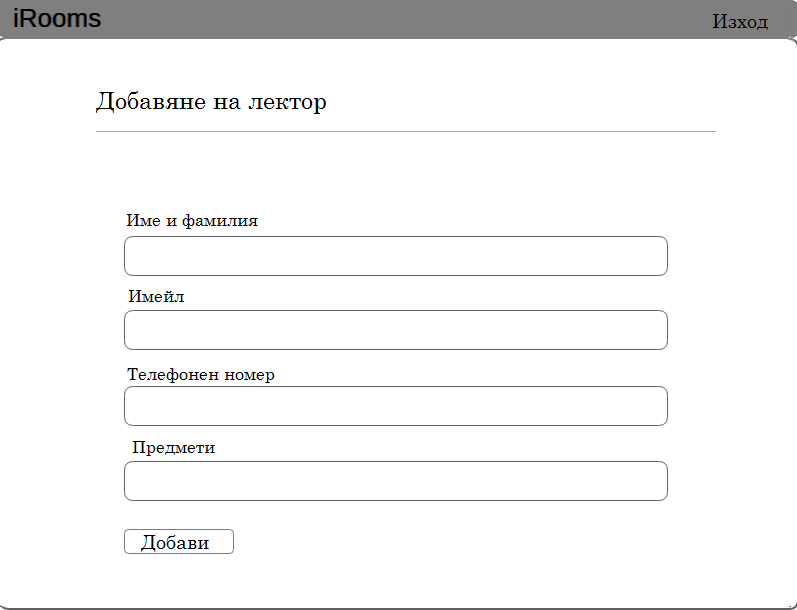
## **User interfaces**











## 

## **Начално описание на нефункционалните изисквания по модела FURPS**

## 

**1.Функционалност(Functionality)**

1.1 Изпращане на нова парола - при забравена парола, системата изпраща нова, произволно генерирана на имейла на потребителя.

1.2 Възможност за допълнително описание на резервация - при резервация на зала, потребителят може да посочи допълнителна информация за това какъв ресурс ще съдържа залата (мултимедия, лаптоп и др.).

**2.Употреба (Usability)**

2.1 Администраторски акаунт – системата поддържа един администраторски акаунт.

2.2 Администраторски интерфейс – администраторът има специален интерфейс, чрез който вижда информация за системата и потребителите и ги администрира.

2.3 Лекторски интерфейс – лекторите имат собствен интерфейс, който им помага да рекуестват за резервация и да следят своите резервации.

2.4 Съвместимост с мобилни устройства – инетрфейсът на системата работи адекватно на различните мобилни устройства (телефони, таблети).

**3.Надеждност (Reliability)**

3.1 Криптиране на пароли – системата поддържа паролите в криптиран вид.

3.2 Устойчивост на данни – няма да се губят данни, въведени преди повече от 24 часа.

3.3 Downtime – системата няма да е в неизправност от повече от 48 часа сборно на година.

3.4 Online backup – системата запазва състоянието си веднъж на всеки 24 часа и продължава да е в изправност докато този процес се изпълнява.

3.5 „Живот“ на backup – можем да се върнем не повече от месец назад.

3.6 „Живот“ на лекторска сесия – лекторските сесии в системата изтичат автоматично при 1 час неактивност на потребителя.

**4.Производителност(Performance)**

4.1 Бързина на зареждане на календара – не повече от секунда и половина.

4.2 Брой посещения – поддържа около 100-150 посещения на ден.

4.3 Брой – активни акаунти – системата поддържа до 1000 активни лекторски акаунта.

**5. Поддръжка(Supportability)**

5.1 Поддръжка на Linux – системата е съвместима с Linux и дъщерните му дистрибуции.

5.2 Жизнен цикъл на системата – 10 години

**6. Ограничения по интерфейса (Interface constraints)**

6.1 Уеб интерфейс – интерфейс достъпен чрез стандартните уеб браузъри (Chrome, Mozilla, IE/Edge, Opera, Safari).

6.2 Мобилен интерфейс – интерфейс, който да функционира адекватно на мобилни устройства.

**7.Ограничения по дизайна (Design constraints)**

7.1 База данни – съхранението на данни се осъществява чрез релационна база данни.

**8.Ограничения по имплементацията (Implementation constraints)**

8.1 Linux платформа – всичко, което правим трябва да е съвместимо и легално в Linux.

## 

## **Пълен списък на потребителските случаи, подредени по реда на тяхната реализация**

## След разработването на основата на системата (core), идентифицираните потребителски случаи ще бъдат реализирани по следния ред:

1. А-2 Вход в системата
2. Б-4 Поискване на резервация на зала от лектор
3. A-1 Поискване на резервация на зала от лектор
4. C-1 Запазване на зала от администратор
5. Б-2 Изтриване на резервация от лектор
6. Б-5 Редактиране на резервация от администратор
7. Б-1 Редактиране на резервация от лектор
8. С-2 Промяна на лични данни
9. С-5 Добавяне на нова стая
10. С-8 Изтриване на лични данни
11. Б-3 Изтегляне на информация
12. С-7 Промяна на парола
13. С-9 Премахване на стая
14. С-6 Добавяне на нова специалност
15. С-10 Редактиране на стая
16. С-11 Премахване на специалност
17. С-12 Редактиране на специалност
18. С-3 Промяна на забравена парола
19. С-4 Изтриване на лектор

9. Примерен план на проекта

9.1 Планиране (Inception)

1. Анализиране на изискванията (1 седмица)

2. Дефиниране на функционалните и нефункционалните изисквания (1 седмица)

3. Формулиране на основните потребителски случаи (1 седмица)

9.2 Детайлизиране (Elaboration)

1. Обща системна архитектура (1.5 седмици)

2. Детайлизиране на потребителските случаи (4 седмици)

3. Проектиране на софтуерната архитектура (2 седмици)

9.3 Изграждане (Construction)

1. Завършване на дейностите по анализ и дизайн (1 седмица)

2. Разработка на ядрото на системата (3 седмици)

3. Разработка и тестване на потребителските случаи, итерация 1 (2 седмици)

- А-2 Вход в системата

- Б-4 Добавяне на лектор

- A-1 Поискване на резервация на зала от лектор

4. Разработка и тестване на потребителските случаи, итерация 2 (2 седмици)

- C-1 Запазване на зала от администратор

- Б-2 Изтриване на резервация от лектор

- Б-5 Редактиране на резервация от администратор

5. Разработка и тестване на потребителските случаи, итерация 3 (2 седмици)

- Б-1 Редактиране на резервация от лектор

- С-2 Промяна на лични данни

- С-5 Добавяне на нова стая

6. Разработка и тестване на потребителските случаи, итерация 4 (2 седмици)

- С-8 Изтриване на лични данни

- Б-3 Изтегляне на информация

- С-7 Промяна на парола

7. Разработка и тестване на потребителските случаи, итерация 5 (2 седмици)

- С-9 Премахване на стая

- С-6 Добавяне на нова специалност

- С-10 Редактиране на стая

8. Разработка и тестване на потребителските случаи, итерация 6 (2 седмици)

- С-11 Премахване на специалност

- С-12 Редактиране на специалност

- С-3 Промяна на забравена парола

- С-4 Изтриване на лектор

9. Системни и интеграционни тестове ( 1 седмица)

9.4 Предаване (Transition)

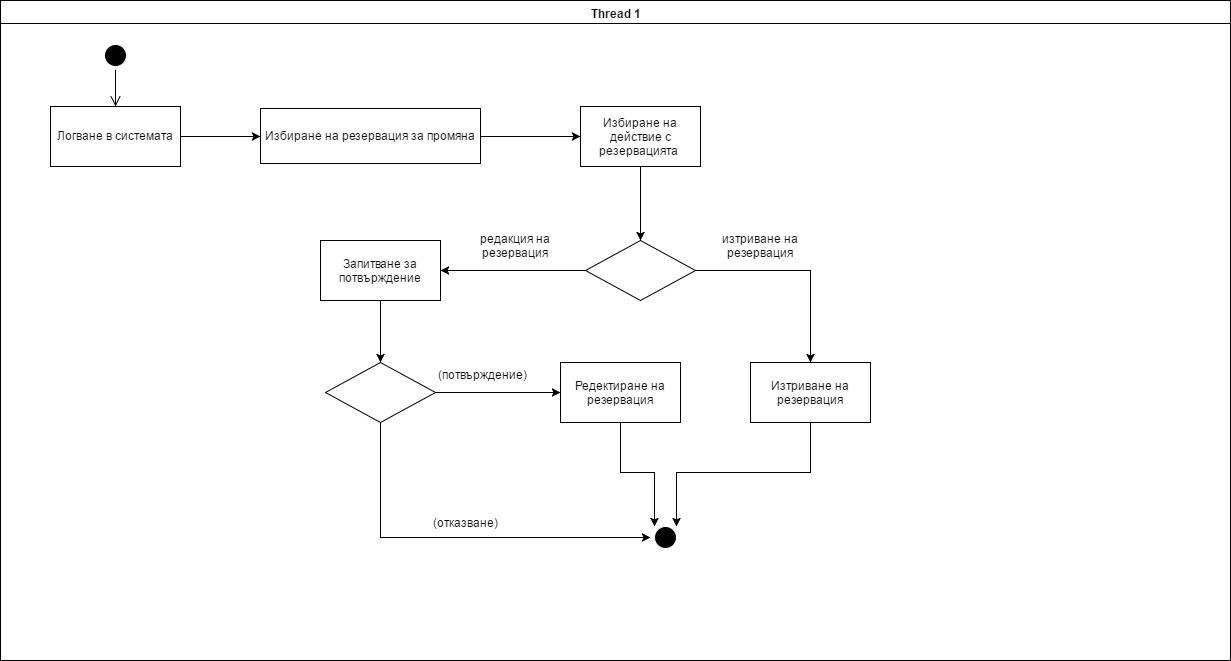
1. Внедряване при клиента (1 седмица)

2. Обучение на потребители (2 дни)

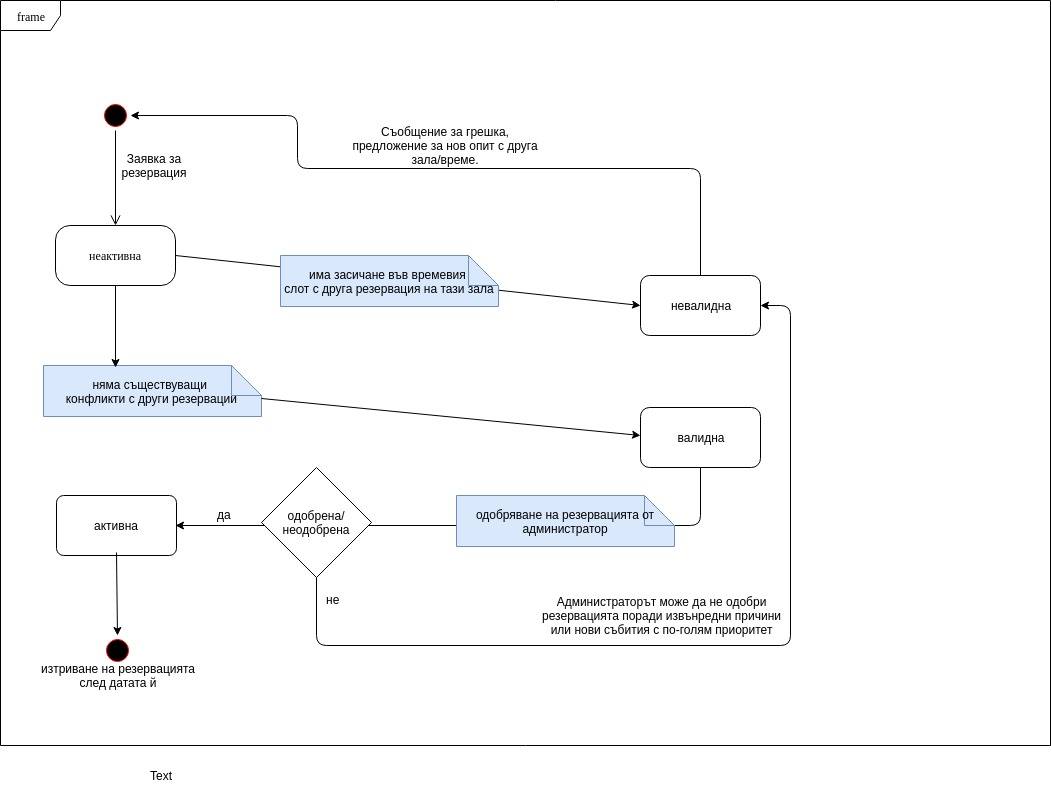
3. Потребителски тестове (1 седмица)

4. Отстраняване на грешки

## **UML диаграми**



След като потребителят се логне в системата, избира резервацията, която иска да промени, като задължително тя трябва да е създадена от него, в противен случай няма право да прави каквито и да било промени. В зависимост от това дали иска да прави редакции или да изтрие резервацията, системата реагира по различен начин. При опит за редакция , като например промяна на часа, залата, или оборудването, системата изпраща запитване до админа и в зависимост от това дали той ще го одобри или отхвърли, съответно промените биват изпълнени, или не. Докато ако потребителят (лекторът) избере да изтрие резервация, тя автоматично бива изтрита, без да има допитване до админа.



При създаване на резервация от страна на лектор, първоначалното и състояние винаги е 'неактивна'. Ако за същата зала и време няма друга направена, състоянието се променя на 'валидна' автоматично от системата. В противен случай резервацията е 'невалидна' и бива показано съобщение за грешка при лектора. Ако тя бъде автоматично валидирана, следва изчакване за одобряване от страна на администратора. В повечето случаи резервацията бива 'одобрена' и променя състоянието си на 'активна', но има и такива, при които се появяват събития с по-голям приоритет и администраторът бива принуден да откаже резервацията.

## **Домейн модел**

## 

## Домейн модела е едно визуално представяне проекта посредством концептуални класове или реални обекти в домейна. Придържайки се към простите правила и внимавайки да не причиним визуален шум ние създадохме нашият домейн модел . В него има различни категории концептуални класове , например "Друга система", "Роли на хора" и други.

## **Речник**

Част 1) Термини от предметната област:  
   
 \*) видео проектор – дигитално устройство, което визуализира снимки или видео материал върху сравнително голяма площ  
  
  
Част 2) Термини със специфично значение в контекста на проекта:

\*) администратор – лице, което отговаря за инсталирането, конфигурирането, управлението, обновяването и поддръжката на определен софтуерен продукт  
   
 \*) профил – е визуално представяне на набор от предоставени лични данни и информация, свързани с конкретен компютърен или онлайн потребител.  
   
  
Част 3) IT термини:  
  
 \*) Интеграция – Обединяване(свързване) на две или повече информационни системи  
  
 \*) Фийчър – част от функционалността на даден софтуерен продукт  
   
 \*) логин – вписване в информационна системата  
   
 \*) акаунт - синоним на профил

## 

## **Време**

|  |  |  |
| --- | --- | --- |
| Студент | Задача | Време |
| Иван Атанасов | 6. Вашата начална визия за бъдещия продукт - функционалност, ограничения, елементи на потребителски интерфейс; изграждане на модели | 4:00 часа |
| Ивелин Тодоров | Обобщение на записките на екипа под формата на описание на наблюденията ни при определяне изискванията; изграждане на модели | 3 часа |
| Богомил Богомилов | Методи за определяне на изискванията и разпределяне на хората от екипа; изграждане на модели | 3 часа |
| Александър Танков | Описване същността на осъществяваната от нас идея; изграждане на модели | 4 часа |
| Валентин Георгиев | Презентацията, изграждането на модели, разпределяне на задачите в екипа | 4:30 часа |
| Иво Райков | Изграждане на модели | 3 часа |
| Георги Димов | Изграждане на модели + презентация | 3:30 часа |

## **Времето за работа на всеки студент по отделните**

## **задачи**

|  |  |  |
| --- | --- | --- |
| Студент | Задача | Време |
| Иван Атанасов | Обсъждане на проекта с колеги, работа върху пропуските от минали път, работа върху изграждането на финалната версия на проекта | 18:00 часа |
| Ивелин Тодоров | Изграждане на user interface-и според създадените use-case-и | 11 часа |
| Богомил Богомилов | Oписание на нефункционалните изисквания по модела FURPS | 8 часа |
| Александър Танков | Събирания и обсъждане на проекта с колеги, обсъждане с отдел студентки, работа върху подобряването на нещата от миналия път и работа по новите задачи | 18 часа |
| Валентин Георгиев | Изграждане на UC\_brief, пълно описание на всички UC, както и приоритизирането им; направа на презентация | 20 часа |
| Иво Райков | Изграждане на модели | 3 часа |
| Георги Димов | Изграждане на модели, презентация,  оформление на документа, провеждане на интервю, | 19:30 часа |



## Тема: „Информационна система за управление на зали”

спец. „Информационни системи“ - 3-курс

Ръководител: проф. д-р Калинка Калоянова

Участници:

Валентин Георгиев ФН: 71563

Георги Димов ФН: 71572

Александър Танков ФН: 71492

Ивелин Тодоров ФН: 71547

Богомил Богомилов ФН: 71591

Иван Атанасов ФН: 71560

Иво Райков ФН: 71561

### **Съдържание:**

    Заглавие и Участници……….……………………..……….…….…...1

Съдържание……………………………………………...…………......2

Методи за определяне на изискванията и разпределение на

хората от екипа…….…………………………………………………...3

    Записки на работата на екипа по определяне на изискванията..3

    Кратко текстово описание на проекта………….……………………4

    Начална визия на продукта: функционалност, ограничения,

елементи на потребителски интерфейс…………………………….4

    Индивидуални модели на работа…………………………………….10

    Use case модел………………………………………………………….17

    Use case шаблон……………………………………………………..…18

    Актьори и роли…………………………………………………………..     Начален списък на потребителските случаи………………………..

    Шаблон за пълно описание на потребителските случаи ……..

    Пълно описание на 10% от потребителските случаи ………...

    Елементи на потребителски интефейс, асоцииран с

    потребителските случаи

    Начално описание на нефункционалните изисквания по модела FURPS

    Речник

    Времето за работа на всеки студент по отделните

    задачи…………………………………………………………....……….15

## 

## Методи за определяне на изискванията и разпределение на хората от екипа

За да определим какво точно иска от нас клиентът ни, ще приложим традиционни методи за определяне на изискванията. Те ще включват добре познатото интервю, което ще представлява среща с клиента очи в очи и дискутиране какви са неговите изисквания, да споделим какви са нашите виждания и какво можем да му предложим. Можем да разглеждаме вече съществуващи системи и да дискутиране с клиента дали му допада нещо конкретно и какво би искал да се заимства от тях като допълнителна функционалност към вече предложената от нас. Интервюто ще е формално, като  ще представлява смесица от отворено и затворено интервю – ще има част с отговори на предварително дефинирани въпроси, както и част с неформално дискутиране на подробности, свързани със системата. За специфичния случай ще е добре да се допитаме и до лекторите, защото и те като хора които ще използват системата , може да има изисквания, които не трябва да се пренебрегват.

## 

## 

## 

## Записки на работата на екипа по определяне на изискванията

     Ще представим записките на екипа под формата на съвкупност от наблюденията ни по време на определяне изискванията за изграждане на системата. Всеки един от нас разгледа различна система, аналогична на нашата, с цел извличане на характеристики, които бихме могли да интегрираме директно или чрез малки корекции и подобрения. Забелязахме как повечето от разгледаните примери в основата си лежат на едни и същи принципи, като разликите бяха или козметични, или леки технически модификации на вече съществуващ такъв. Предвид опита на всеки един от нас, решихме да заложим на добре изпитан модел, в който се акцентира върху гъвкавост и надеждност. Първоначалното ни виждане за създаването на системата претърпя известна промяна след разглеждането на вече съществуващи подобни имплементации, което донякъде ни помогна и да дефинираме изискванията.

В последствие решихме да се консултираме с отговорното лице на ФМИ и да разберем как точно седят нещата в конкретната организация. След задаване на поредица от въпроси успяхме да добием подробна представа за това, как работи системата в университета.При изготвянето на графика лекторите не участват пряко в избора на ден, час или зала. Само когато някой лектор по- взискателен, тогава се комуникира с него и се взима и неговото мнение под въпрос.

## 

## Кратко текстово описание на проекта

Проекта „Информационна система за управление на зали” е свързан с изграждането на платформа, която има за цел да улесни процеса около разпределянето на зали в различни учреждения като университети, училища и др. Тя ще предоставя удобен интерфейс на клиента, чрез който той лесно ще може да види даден календар (на стая,специалност, лектор и др.), да добавя различни изисквания при разпределянето на залите или да ги запазва ръчно. Потребители на системата ще бъдат преподавателите и хората, занимаващи се с разпределянето на залите преди сесия или семестър.

## Начална визия на продукта: функционалност, ограничения, елементи на потребителски интерфейс

### **Функционалности:**

* Запитване за запазване на зала от лектор
* Вход в системата
* Редактиране на резервация от лектор.
* Изтриване на резервация от лектор.
* Промяна на лични данни
* Промяна на забравена парола.
* Изтегляне на информация
* Добавяне на лектор
* Изтриване на лектор
* Добавяне на нова стая.
* Добавяне на нови специалности.
* Преглед на детайлите на системата.
* Редактиране на резервация от администратор
* Запазване на зала от администратор
* Промяна на парола
* Редактиране на лектор от администратора
* Премахване на стая
* Редактиране на стая
* Премахване на специалност
* Редактиране на специалност
* Интеграция със система, ползваща се за генериране на учебния график

### **Системата трябва да поддържа следните нефункционални изисквания:**

* Да е достъпна през уеб интерфейс
* Да поддържа около 100-150 посещения на ден
* Да поддържа 1 администраторски акаунт
* Да поддържа 1000 лекторски акаунта
* Да осигурява зареждане на календара до 1-1.5 секунди
* Да поддържа паролите в криптиран вид
* Да може да е в експлоатация 10 години
* Лекторските сесии в системата изтичат автоматично при 1 час неактивност на потребителя
* Да има автоматичен онлайн бекъп не по-рядко от веднъж на 24 часа

В основата си, нашата система  е специфично насочена към университета и по-конкретно, към ФМИ. Тя ще предоставя точен график на задължителните предмети от текущия семестър, както и на избираемите предмети. Преди сесия ще могат да бъдат видими и графиците за изпитите, къде ще се състоят и от колко часа.

Съставянето на графиците за семестриалната година ще бъде извършена в няколко етапа:

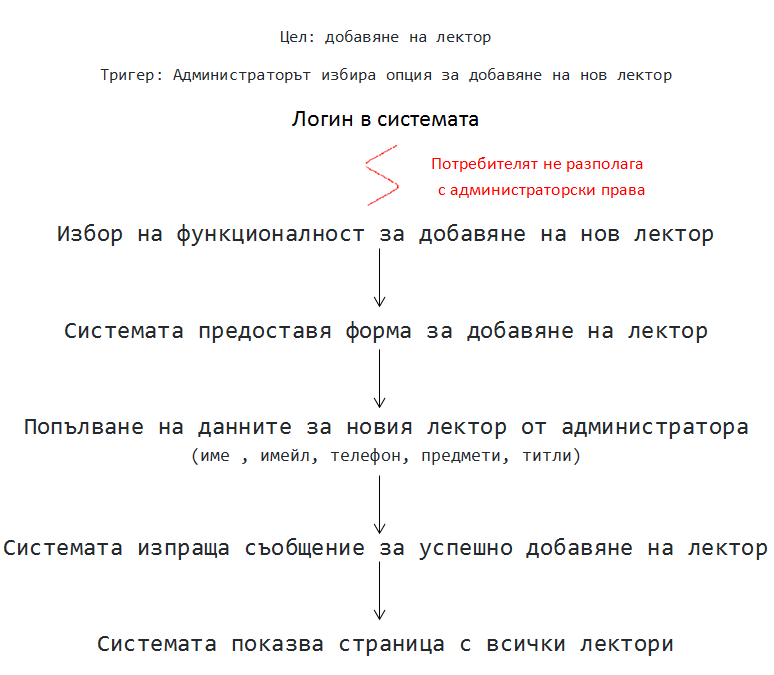
1. В 10 дневен срок лекторите, които имат предпочетания относно деня, датата и мястото на тяхното занятие ще имат възможност да попълнят форма.
2. След 10 дневния срок администратора, комуникира с лекторите (ако е нужно).
3. За останалите занятия ,където не е имало активност на лектор,  на случаен принцип се генерират събития, които се въвеждат в календара.
4. След стъпка 1, 2, 3,  календарите на лекторите, стаите и специалностите вече са изготвени за текущия семестър.

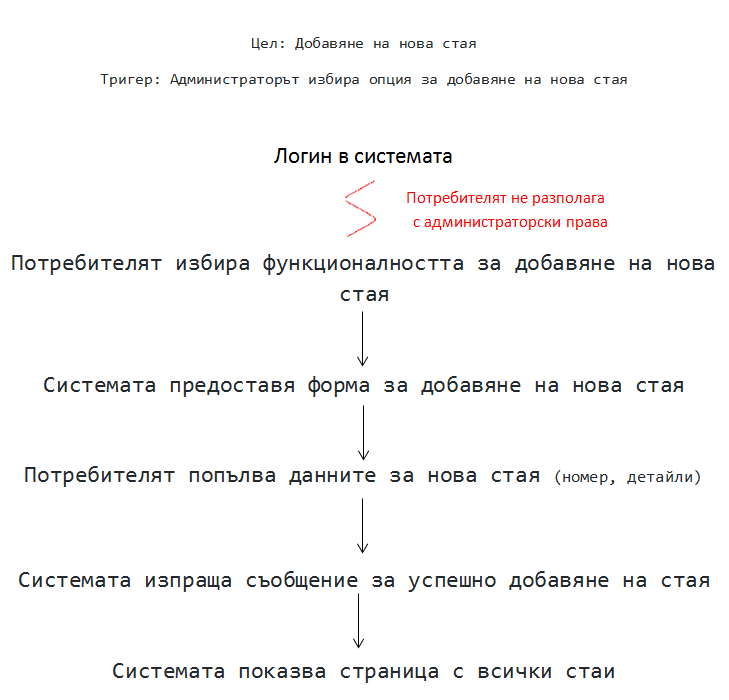
1. За избираемите курсове, допълнителните упражнения и консултации  ще се разчита на всеки лектор да е попълнил формата за график, след което администратора да ги е одобрил.
2. Преди сесия, всеки лектор трябва в 10 дневен срок да въведе своите предпочетания, които са съобразени със студентите.
3. След 10 дневният срок администратора допълнително комуникира с лекторите( ако е нужно ).
4. Графика е изготвен спрямо заявленията на лекторите.

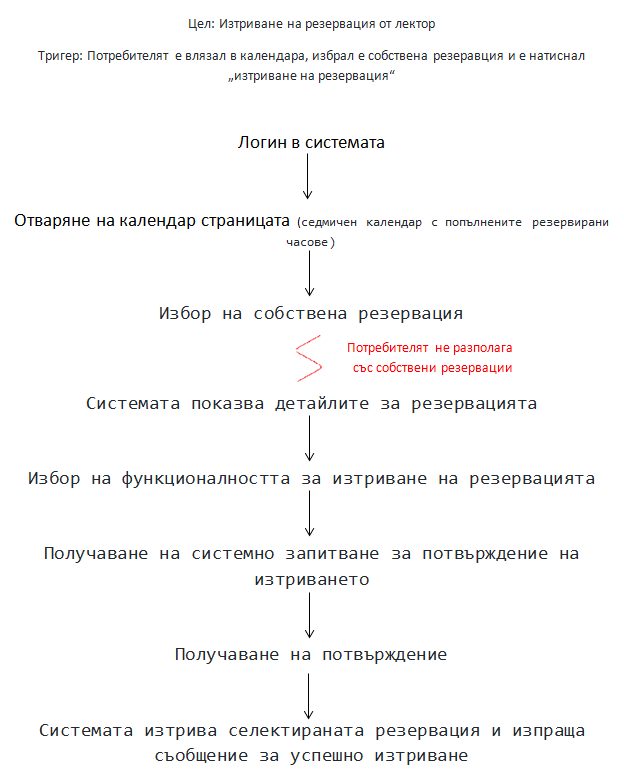
Учебния график  ще представлява съвкупност от записи, които ще съдържат в себе си следните атрибути: Title, Room, Date, Duration, Start/end time, specialty. Ще се предлага функционалност, чрез която да се преглеждат календари за специалността, на избран лектора или на определена стая. Трите календара ще се съобразяват един с друг, така че да не се получава застъпване на заниманията в календарите. Поради факта че консултациите се извършват веднъж, ще има фунционалност, чрез която единично да се добавя събитие в календара. Дадено събитие ще може да се конфигурира, така че да се повтаря през определен период от време(сесия) или да съществува за избраната дата.

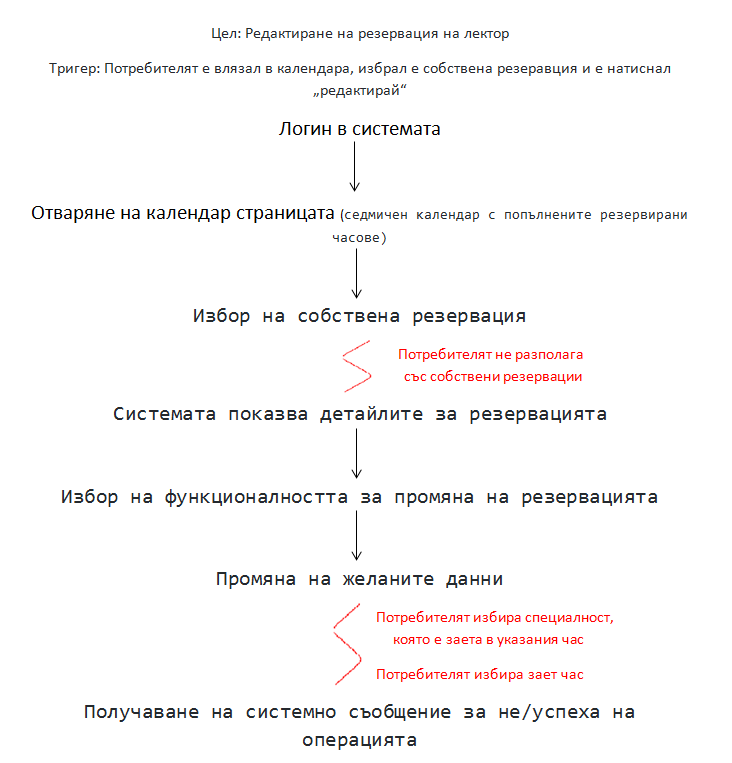
При създаването на стая администратора задава номер и тип на стаята - дали е за лекции или упражнения( с компютърна база). Друга доста полезна функционалност ще бъдe възможността да се добавят ресурси  към съответна стая. Администратора ще може да конкретизира дали в стаята има видео проектор, бяла дъска или екран за презентиране. Всяка стая трябва да притежава атрибут Capacity, за да се избегне случай, в който стаята е твърде малка за специалността. Всяка стая има собствен календар, който може да бъде изтеглен, като отделен ресурс.

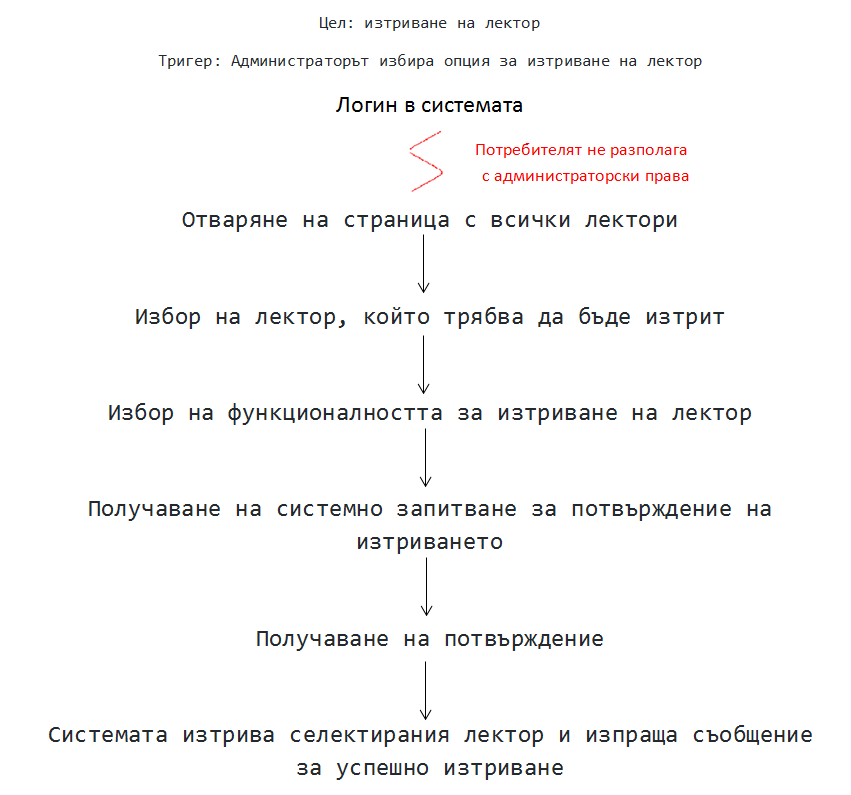
## Индивидуални модели на работа

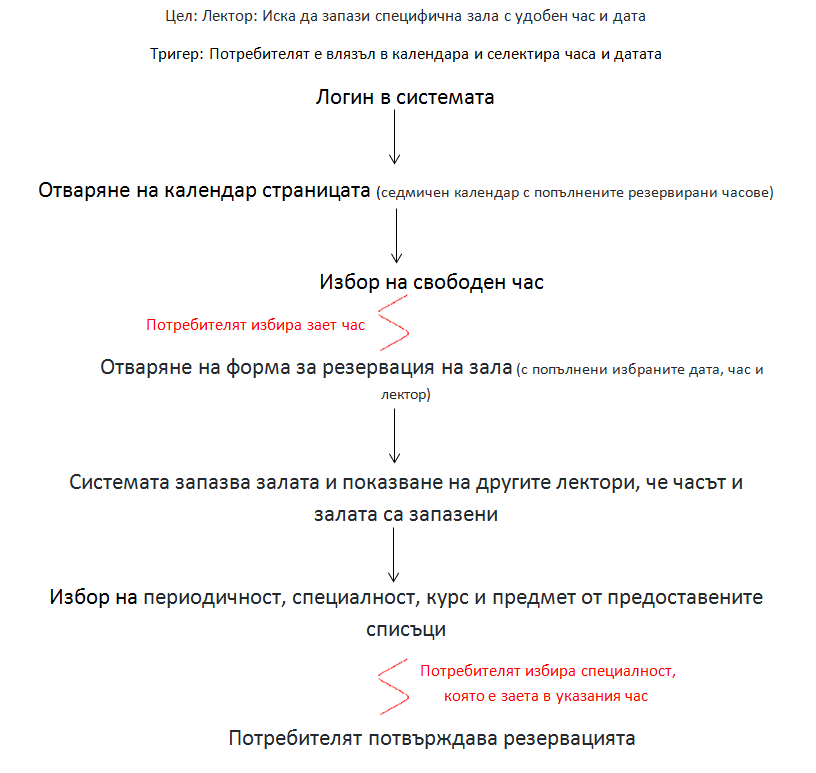


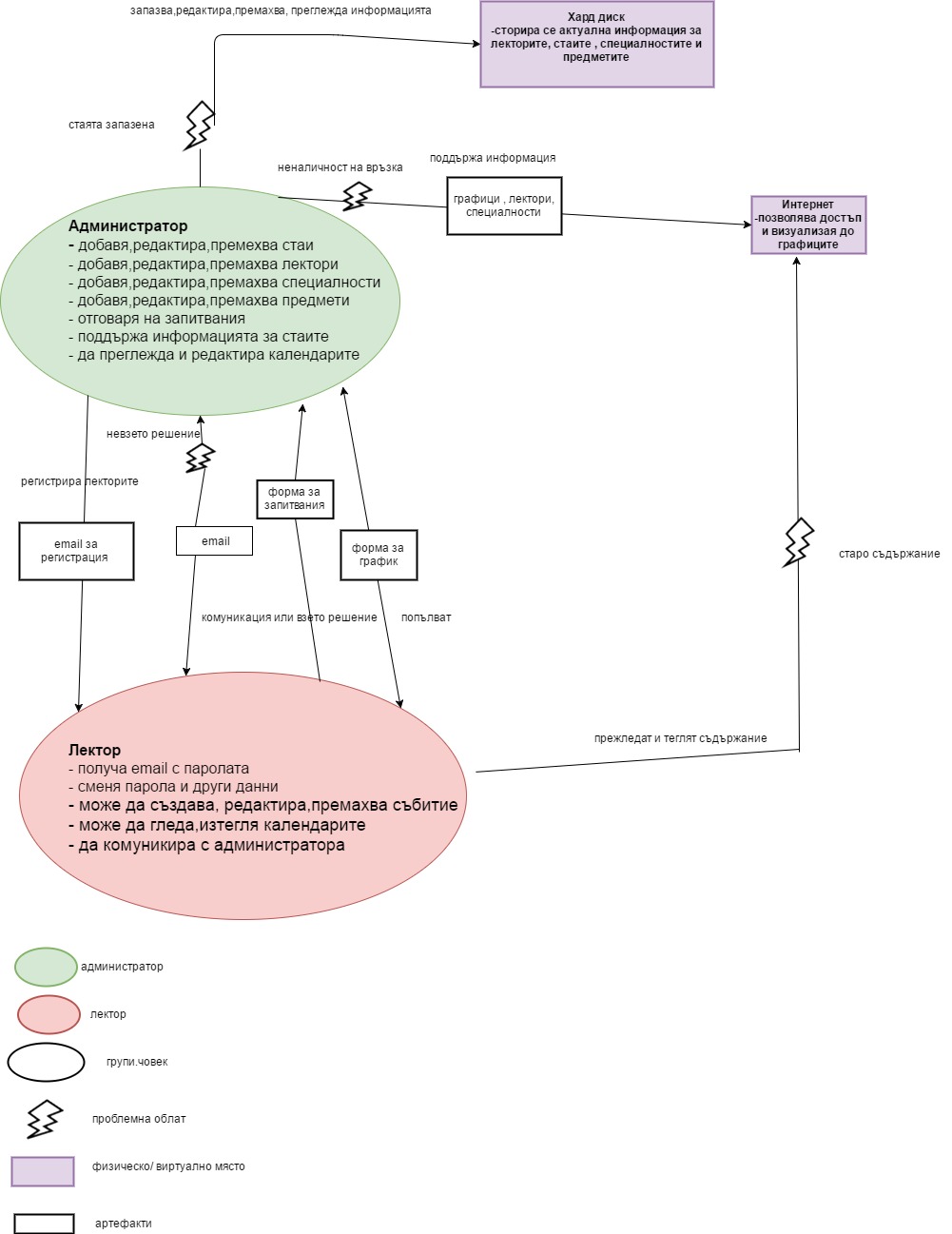








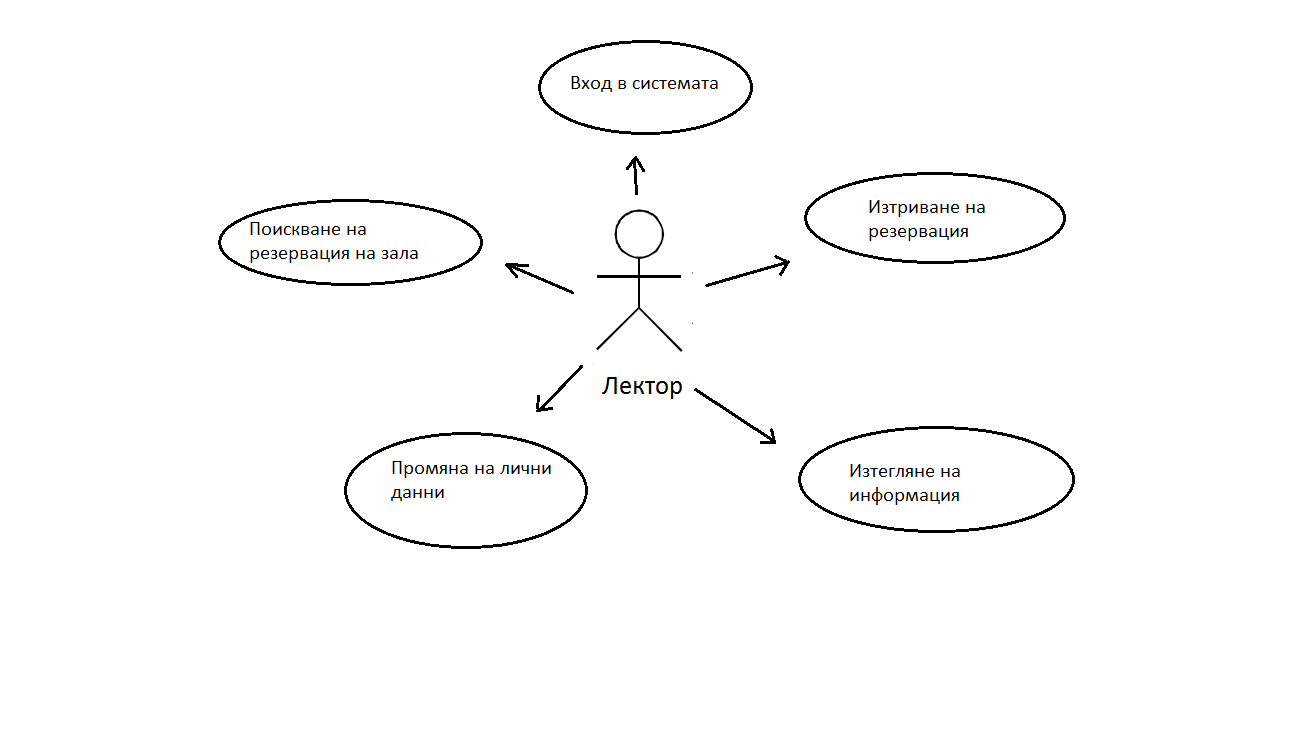
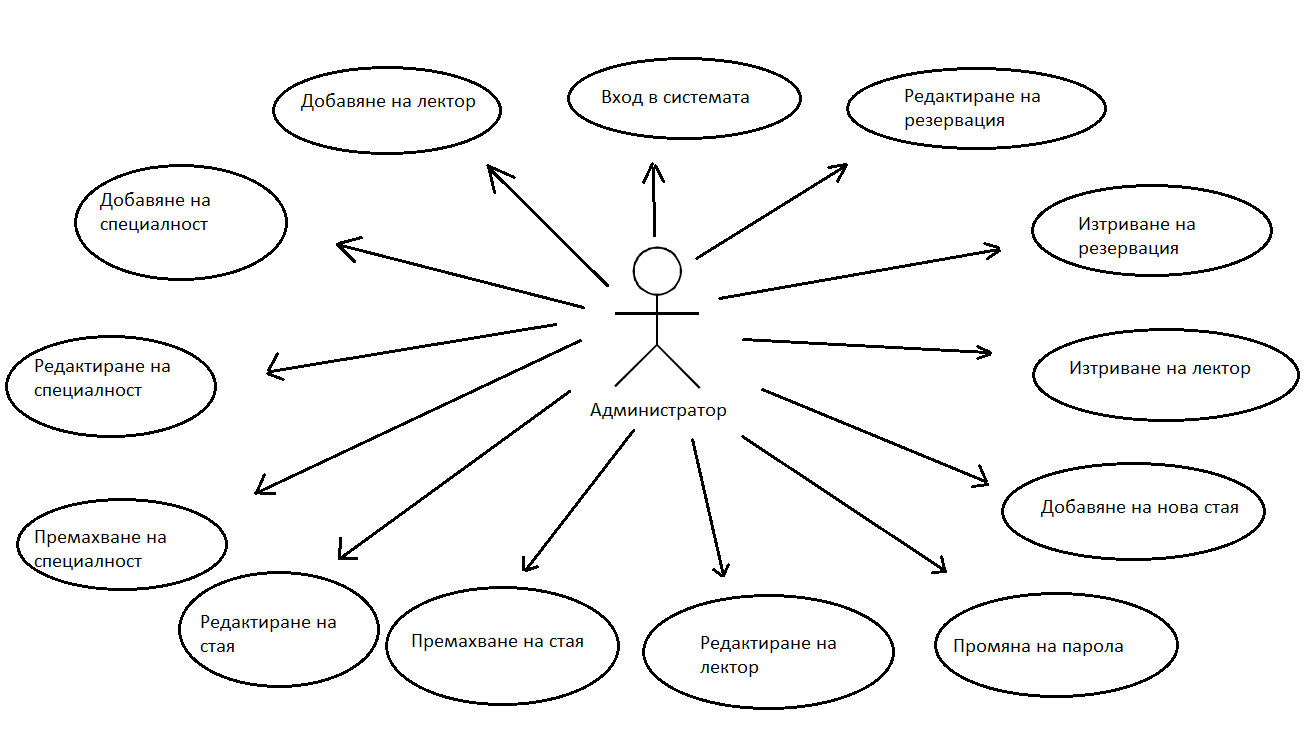




## 

## Use case модел

Mодела на потребителските случаи показва системните изисквания в контекста на потребителския случай.

Като актьор на модела е представен Лектора, който може да попадне в няколко от посочените потребителски случаи. Също може да се разгледа модела с актьор Администратор. Там имаме значително повече потребителски случаи. 

## Актьори и роли

Системата трябва да поддържа следните актьори:

* Лектор:
  + Който трябва да може:
    - Да се вписва в системата
    - Да проверява за празна стая
    - Да прави запитване за резервация на стая
    - Да редактира и изтрива собствена резервация
    - Да може да си изтегля някаква информация (например календари) в pdf формат
    - Да променя личните си данни (email, телефон, предмети)
    - Да промяна паролата си
* Администратор:
  + Който трябва да може:
    - Да има всичките права на лектора
    - Да създава, редактира и изтрива акаунти на лектори
    - Да разглежда личните данни на лекторите, въведени в системата
    - Да добавя, редактира, изтрива нови стаи
    - Да добавя, редактира и изтрива нови специалности
    - Да вижда списък с лекторите, стаите и специалностите в системата
    - Да изпраща имейл с покана до лекторите на факултета

Ролите, като част от домейн модела трябва да са следните:

* Стая – всяка една стая в университета, която може да се ползва, ще бъде добавена в системата от администратора. Така лекторите ще могат лесно да запазват различни стаи в зависимост от техните изисквания. Всяка стая ще съдържа следната информация в себе си:
  + Тип (с мултимедия, с компютри и т.н)
  + Номер
  + Капацитет
  + Заетост – Това ще представлява собствен календар на стаята, в който може да се провери тази стая кога е свободна и кога не. Ако е запазена, ще има информация кой я е запазил, за коя специалност и какъв точно е предметът.
* Специалност:
  + Година
  + Брой хора (като може лесно да се вземе от броя на хората в групите)
  + Групи – Представлява различните групи от хора в специалността като всяка една група ще съдържа следната информация:
    - Номер
    - Брой хора
    - Учебен план – Той описва какви точно ще са задължителните
* Предмети
  + Седмична заетост (колко на брой лекции и упражнения ще имат за една седмица по даденият предмет)
  + Лектор – Кой ще я води
  + Семестър(зимен/летен) – в кой курс и семестър ще се изучава
  + Курс и специалност (за задължителните предмети)

* Резервация
  + Име – Когато се избира за даден предмет то ще бъде автоматично генериран и ще съдържа името на специалността, името на предмета и името на лектора. Ако не се избира за даден предмет (например за консултация) авторът на резервацията попълва името.
  + Кога – в кой ден и кой час
  + Къде – в коя стая
  + Допълнителна информация

## Use case шаблон

use case <номер на UC>: <име на UC>

scope:  ...

level: user-goal или subfunction

primary actor: ...

stackeholders & interesets:

            - един: какво е важно за него

            - друг: ...

trigger:

preconditions: (какво трябва да е изпълнено преди началото и си заслужава да го кажем)

            - едно

            - друго

postconditions: (какво трябва да е вярно след успешен завършек и си заслужава да го кажем)

            - едно

            - друго

main success scenario: (типичния, безусловен "хепи" път на успешен сценарий)

            1. действие

            2. действие

            ...

extensions: (алтернативни сценарии (успешни/не))

            2a. грешен логин

                           1. нещо

                           2. нещо

            5а. друга алтернатива

                           1. нещо

                           2. нещо

special requirements:

            - един

            - друг (нефункционално изискване)

frequency of occurrence: (оказва влияние на това кога ще задълбаем в проучването, реализацията, тестването и т.н.)

            веднъж на пълнолуние

други:

            забележки, въпроси

## 

## **Екипно решихме да приоритизираме всички пълни описания на потребителските случаи по това колко често ще се използват в информационната система за разпределение на зали**

# A.

1. **Поискване на резервация на зала от лектор**

Всеки лектор в системата има право да поиска да запази зала. Когато той избере час, периодичност и зала, трябва да попълни заглавие на заниманието, което ще се провежда, курс и специалност. В резултат системата изпраща до администратора поискването на лектора и той преценява дали да запази тази зала за него или не.

use case 1: Поискване на резервация на зала от лектор

scope: N/A

level: user-goal

primary actor: Лектор

stakeholders & interesets:

   Лектор: Иска да запази специфична зала с удобен час и дата

trigger:

   Потребителят е влязъл в календара и селектира часа и датата.

preconditions:

   Лекторът трябва да е влязъл в системата

postconditions:

   Инициаторът е поискал да запазил успешно залата и се чака отговор от администратора.

main success scenario:

   01. Потребителят отива на календар страницата

   02. Системата предоставя седмичен календар с попълнени резервираните часове

   03. Потребителят избира свободен час.

   04. Системата предоставя форма за резервация на зала с попълнени избраните дата, час и лектор.

   05. Потребителят избира периодичност, специалност, курс и предмет от предоставените списъци.

   06. Потребителят потвърждава резервацията.

   07. Системата изпраща поискването на лектора за резервация към администратора.

   extensions:

   03a. Потребителя избира зает час

1. Системата информира потребителя, че няма как да запази вече зает час.

2. Продължаваме със стъпка 03 от главния сценарий.

   06а. Потребителя избира специалност, която е заета в указания час.

       1. Системата информира потребителя, че специалноста е заета.

2. Продължаваме от стъпка 03 от главния сценарий.

   07а. В продължение на дълъг период от време потребителят не потвърждава резервацията.

1. Системата премахва резервацията на стаята за този час

2. Продължаваме със стъпка 02 от главния сценарий.

special requirements:

   - системата запазва залата за по-малко от 0.8 секунди.

frequency of occurrence:

   - Винаги, когато лектор иска да запази зала.

други:

   N/A

**2.  Вход в системата**

Всеки потребител на информационната система трябва да може да влезе в нея. След успешно влизане в системата, в зависимост от ролята, потребителят има различни възможности и права.

use case 2: Вход в системата

scope: N/A

level: user-goal

primary actor: лектор, администратор

stakeholders & interesets:

   - Лектор: Иска да влезе в системата

   - Администратор: Иска да влезе в системата

trigger:

   Потребителят е достъпил системата като гост.

preconditions:

   Потребителят да е заредил информационната система и тя е предоставила форма за вход в системата

postconditions:

   Потребителят е попълнил правилно формата и получава достъп до системата.

main success scenario:

   01. Потребителят попълва потребителско име и парола

   02. Потребителят избира функционалност за вход

   03. Системата проверява валидността на данните

   04. Системата предоставя достъп до функционалностите в зависимост ролята

   extensions:

   03a. Потребителя попълва грешни данни

       1. Системата показва съобщение за грешни входни данни.

       2. Продължаваме със стъпка 01 от главния сценарий.

special requirements:

   - системата проверява и дава достъп на потребителя за по-малко от 1.4 секунди.

frequency of occurrence:

   - Винаги, когато потребител иска да влезне в системата с акаунта си.

други:

   N/A

# B.

**1. Редактиране на резервация от лектор.**

Лекторът, който е запазил зала, има възможност да променя детайлите по тази резервация. Това се прави с цел ако трябва да се променят часа или залата за занятието. В резултат на промяната има проверка дали залата е свободна за указаната дата. Ако е свободна се променя резервацията, ако не е се съобщава на потребителя.

use case 3: Редактиране на резервация от лектор

scope: N/A

level: user-goal

primary actor: лектор

stakeholders & interesets:

   - Лектор: Иска да промени данните на резервация, която сам е направил

trigger:

   Потребителят е влязъл в календара, избрал е собствена резервация и е натиснал редактиране на резервация

preconditions:

   Лекторът трябва да е влязъл в системата

   Лекторът трябва да има поне една резервация

postconditions:

   Инициаторът е променил успешно данните на резервацията.

main success scenario:

   01. Потребителят отива на календар страницата

   02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

   03. Потребителят избира собствена резервация

   04. Системата показва детайлите на резервацията

   05. Потребителят избира функционалноста за промяна на резервацията

   06. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията с изключение на датата и часа.

   07. Потребителят променя данните, които иска да промени

   08. Системата проверява дали указаните данни са правилни

   09. Системата показва съобщение за успешно променена резервация

   extensions:

   08а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 07 от главният сценарий

special requirements:

   - системата променя резервацията за по-малко от 1 секунда.

frequency of occurrence:

   - Винаги, когато лектор иска да редактира резервация.

други:

   N/A

**2. Изтриване на резервация от лектор.**

Лекторът, който има запазена зала, има възможност да премахне резервацията и да освободи залата. Целта е, ако лекторът не може да присъства на заниманието да го отмени и освободи залата.

use case 4: Изтриване на резервация от лектор

scope: N/A

level: user-goal

primary actor: лектор

stakeholders & interesets:

   Лектор: Иска да изтрие резервация, която сам е направил

trigger:

   Лекторът е влязъл в календара, избрал е собствена резервация и е натиснал изтриване на резервация

preconditions:

   Лекторът трябва да е влязъл в системата

   Лекторът трябва да има поне една резервация

postconditions:

   Лекторът е изтрил успешно резервацията.

main success scenario:

   01. Потребителят отива на календар страницата

   02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

   03. Потребителят избира собствена резервация

   04. Системата показва детайлите на резервацията.

   05. Потребителят избира резервацията, която иска да изтрие.

   06. Системата пита за потвърждение за изтриването на резервацията.

   07. Потребителят потвърждава изтриването на резервацията.

   08. Системата изтрива резервацията

   09. Системата изпраща съобщение за успешно изтриване.

   extensions:

    07а. Потребителя отказва изтриване на резервация

    - Продължаваме със стъпка 04 от главния сценарий

special requirements:

   - системата изтрива резервацията за по-малко от 0.7 секунда.

frequency of occurrence:

   - Винаги, когато лектор иска да изтрие собствена резервация.

други:

   N/A

**3. Изтегляне на информация**

Всеки потребител има възможност да изтегли информация за графика си по месец/седмица/специалност/курс в pdf формат. След успешно изтегляне, лектора може да разпечата графика си или да го изпрати на студенти.

use case 7: Изтегляне на информация

scope: N/A

level: user-goal

primary actor: лектор, администратор

stackeholders & interesets:

   - Лектор: Може да изтегли графика си със всички запазени от него зали за определен период от време.

   - Администратор: Може да изтегли графика на всеки лектор, зала, специалност и курс за определен период от време.

trigger:

   Лектор/Администратор избира опция изтегляне на график

preconditions:

   Потребителят е влязъл в системата като лектор/администратор и е влязъл в профила си.

postconditions:

   Изтегля график успешно за избрания период от време.

main success scenario:

01. Потребителя отива на изтегляне на график.

02. Системата предоставя опции за изтегляне на график (администратор: курс/специалност/стая/лектор и период от време: месец, седмица) (лектор: период от време: месец/седмица).

  03. Потребителя попълва данните, за които иска да изтегли информация

  04. Системата изтегля графика

extensions:

 N/A

special requirements:

 Системата изтегля графика за около 10 секунди

frequency of occurrence:

 Всяка седмица или месец от лектор/администратор

други:

 N/A

**4. Добавяне на лектор**

Администраторът има права и възможност да добавя нови лектори към факултета. Ако нов колега бъде назначен на работа, да може администраторът лесно и удобно да му създаде акаунт с който лектора да има достъп до системата.

use case 8: Добавяне на лектор

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

   - Администратор: Може да добавя нов лектор, ако бъде назначен такъв.

trigger:

   Администратор избира опция за добавяне на нов лектор

preconditions:

   Потребителят е влязъл в системата като администратор

postconditions:

   Добавен е нов лектор успешно и е показана страницата с всички лектори

main success scenario:

  01. Потребителят отива на добавяне на нов лектор

  02. Системата предоставя форма за добавяне на лектор.

  03. Потребителят попълва данните за новия лектор (име, имейл, телефон, предмети, титли)

  04. Системата валидира попълнените данни

  05. Системата изпраща съобщение за успешно добавяне на лектор

  06. Системата показва страница с всички лектори.

extensions:

  04а Валидацията на данните е неуспешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 02 от главният сценарий

special requirements:

   Системата добавя нов лектор за около 1 секунда

frequency of occurrence:

   Всяка година по няколко пъти ще се използва от администратор

други:

   N/A

**5. Редактиране на резервация от администратор**

Лекторът, който е запазил зала, има възможност да променя детайлите по тази резервация. Това се прави с цел ако трябва да се променят часа, периодичността или залата за занятието. В резултат на промяната има проверка дали залата е свободна за указаната дата. Ако е свободна се променя резервацията, ако не е се съобщава на потребителя.

use case 14: Редактиране на резервация от администратор

scope: N/A

level: user-goal

primary actor: администратор

stakeholders & interesets:

   - Администратор: Иска да промени данните на резервация

trigger:

   Потребителят е влязъл в календара на дадена специалност/стая, избрал е резервация и е натиснал редактиране на резервация

preconditions:

   Администраторът трябва да е влязъл в системата

postconditions:

   Администраторът е променил успешно данните на резервацията.

main success scenario:

   01. Потребителят отива на календар страницата на специалност/стая

   02. Системата предоставя седмичен календар с попълнени резервираните часове

   03. Потребителят избира резервация

   04. Системата показва детайлите на резервацията.

   05. Потребителят избира функционалноста за промяна на резервацията.

   06. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

   07. Потребителят променя данните, които иска да промени.

   08. Системата проверява дали указаните данни са правилни (проверява дали часа и залата са сменени с други и дали са свободни и дали лекторът е свободен в този час )

   09. Системата информира лектора за промянта

   09. Системата показва съобщение за успешна промяна

   extensions:

08а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на резервацията.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 07 от главният сценарий

special requirements:

   - системата променя резервацията за по-малко от 1 секунда.

frequency of occurrence:

   - Винаги, когато администратор иска да редактира резервация.

други:

   N/A

# C.

**1. Запазване на зала от администратор**

Всеки администратор в системата има право да запази зала. Когато той избере час и зала, трябва да избере лектора, който ще я води. Също така трябва да попълни заглавие на заниманието, което ще се провежда, курс и специалност. В резултат системата запазва залата и блокира възможноста да се правят други резервации за резервирания час и място от други лектори.

use case 13: Запазване на зала от администратор

scope: N/A

level: user-goal

primary actor: администратор

stakeholders & interesets:

   - Администратор: Иска да запази специфична зала с удобен час и дата и лектор

trigger:

   Потребителят е влязъл в календара и селектира часа и датата.

preconditions:

   Администраторът трябва да е влязъл в системата

postconditions:

   Администраторът е запазил успешно залата и тя е резервирана за дадения час.

main success scenario:

   01. Потребителят отива на календар страницата за дадения лектор

   02. Системата предоставя седмичен календар на лектора с попълнени резервираните часове

   03. Потребителят избира свободен час.

   04. Системата предоставя форма за резервация на зала с попълнени избраните дата и час.

   05. Потребителят избира зала, лектор, периодичност, специалност, курс и предмет от предоставените списъци.

   06. Системата проверява данните

   07. Системата запазва залата и показва на другите лектори, че часът и залата са запазени.

   extensions:

   03a. Потребителя избира зает час

       01. Потребителя няма възможност да запази вече зает час.

02. Продължаваме със стъпка 02 от главният сценарий

   06а. Потребителя избира специалност, която е заета в указания час.

       01. Системата изпраща съобщение на потребителя, че специалноста е заета.

02. Продължаваме със стъпка 02 от главният сценарий

special requirements:

   - системата запазва залата за по-малко от 0.8 секунди.

frequency of occurrence:

   - Често използвана функционалност от администратор

други:

   N/A

**2. Промяна на лични данни**

Всеки лектор има възможност да промени личните си данни. Да добави титла, редактира парола, да смени имейл, телефон или да добави предметите, които провежда. В резултат на това, всеки друг потребител ще види новите данни на лектора.

use case 5: Промяна на лични данни

scope: N/A

level: user-goal

primary actor: лектор

stakeholders & interesets:

   - Лектор: Иска да промени личните си данни (титли, телефон, имейл или да добави предмети, които ръководи)

trigger:

   Потребителят е влязъл в системата и натиска профил.

preconditions:

   Лекторът трябва да е влязъл в системата

postconditions:

   Инициаторът е променил успешно личните си данни.

main success scenario:

   01. Потребителят отива на профил страницата

   02. Системата предоставя личните данни, които има лектора

   03. Потребителят променя данните, които желае.

   04. Системата валидира данните

   05. Системата показва съобщение за успешно променени данни.

   extensions:

   04a. Валидацията на данните е неуспешна

       01. Системата показва съобщение за грешка свързано с невалидните данни

02. Системата предоставя личните данни, които има лектора

03. Системата сигнализира кои полета са попълнени грешно

04. Продължаваме със стъпка 03 от главния сценарий

special requirements:

   - системата променя данните на лектора за по-малко от 1 секунда.

frequency of occurrence:

   - Винаги, когато лектор иска да промени личните си данни.

други:

   N/A

**3. Промяна на забравена парола.**

Ако потребителя е забравил паролата си, има възможност да я поднови с нова на ниво вход в системата. Новата парола се генерира динамично от системата и се изпраща на имейла на лектора. След успешно влизане в системата с новата парола, лекторът има възможност да я замени с друга парола.

use case 6: Промяна на забравена парола

scope: N/A

level: user-goal

primary actor: гост

stakeholders & interesets:

   - Гост: Ако потребителя е забравил паролата си и иска да я възстанови.

trigger:

   Потребителят е забравил паролата си и иска да я смени.

preconditions:

   Потребителят е достъпил системата като гост

postconditions:

   Потребителят е получил успешно новата парола.

main success scenario:

   01. Потребителят избира функционалността за забравена парола

   02. Системата показва форма, която съдържа поле за въвеждане на имейл, на който ще бъде изпратена новата парола.

   03. Потребителят въвежда своя имейл.

   04. Системата проверява имейла дали съществува в базата от данни

   05. Системата генерира новата парола

   05. Системата изпраща новата парола

   extensions:

   04a. Имейла, който е въведен не е намерен в базата от данни

01. Системата показва съобщение за грешка

02. Продължаваме със стъпка 02 от главния сценарий.

special requirements:

   - системата изпраща нова динамично генерирана парола на имейла на потребителя за по-малко от 5 секунда.

frequency of occurrence:

   - Винаги, когато лектор или администратор си е забравил парилата и иска да я смени.

други:

   N/A

**4 Изтриване на лектор**

Администраторът има възможност да премахне лектор от списъка с действащи лектори. Това се прави с цел, ако даден лектор напусне или се пенсионира.

use case 9: Изтриване на лектор

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

   Администратор: Може да изтрива лектор, ако бъде уволнен или пенсиониран.

trigger:

   Администраторът отива на страница, на която са показани всички лектори и избира опция за изтриване на лектор

preconditions:

   Потребителят е влязъл в системата като администратор

postconditions:

   Изтриване на лектор успешно от администратора

main success scenario:

 01. Потребителят отива на страницата, на която са показани всички лектори

 02. Системата показва всички лектори.

 03. Потребителят избира лектор, когото иска да изтрие.

 04. Потребителят избира функционалността за изтриване на лектор

 05. Системата изпраща системно запитване за потвърждение на изтриването

 06. Потребителят потвърждава изтриването

 07. Системата изтрива лектора

 08. Системата изпраща съобщение за успешно изтриване на лектор

extensions:

   N/A

special requirements:

   Системата изтрива лектор за около 0.5 секунди.

frequency of occurrence:

   Всяка година по няколко пъти ще се използва от администратор

други:

 N/A

**5. Добавяне на нова стая.**

Администраторът може да добавя нова стая към списъка със стаи. Целта е, когато има разширяване на факултета да може да се добавят лесно и удобно нови стаи с техните детайли.

use case 10: Добавяне на нова стая

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

   - Администратор: Може да добавя нова стая, ако бъде присвоена/построена такава.

trigger:

   Администратор избира опция добавяне на нова стая

preconditions:

   Потребителят е влязъл в системата като администратор

postconditions:

   Добавена е нова стая успешно и страницата със стаите е визуализирана

main success scenario:

  01. Потребителят отива на добавяне на нова стая

  02. Системата предоставя форма за добавяне на стая.

  03. Потребителят попълва данните за нова стая (номер, детайли)

04. Системата изпраща съобщение за успешно добавяне на стая

  05. Системата показва страница с всички стаи.

extensions:

   N/A

special requirements:

 Системата добавя нова стая за около 1 секунда

frequency of occurrence:

Един път на няколко години от администратор

други:

 N/A

**6. Добавяне на нови специалности.**

Администраторът има възможност да добавя нови специалности. В резултат на това списъка със специалности, които лекторите виждат като запазват стаи ще е напълно актуален.

use case 11: Добавяне на нови специалности

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

- Администратор: Може да добавя нова специалност, ако бъде създадена такава.

trigger:

   Администратор избира опция добавяне на нова специалност

preconditions:

   Потребителят е влязъл в системата като администратор

postconditions:

   Добавена е нова специалност успешно

main success scenario:

  01. Потребителят отива на добавяне на нова специалност

  02. Системата предоставя форма за добавяне на специалност.

  03. Потребителят попълва данните за нова специалност (курсове, капацитет, предмети)

       04. Системата валидира данните

  05. Системата изпраща съобщение за успешно добавяне на специалност

  06. Системата показва страница с всички специалности.

extensions:

  04а Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на специалността

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 03 от главният сценарий

special requirements:

   Системата добавя нова специалност за около 1 секунда

frequency of occurrence:

   Един път на няколко години от администратор

други:

 N/A

**7. Промяна на парола**

Ако потребителят иска да смени паролата си, има възможност да го направи след като се е вписал в системата. От полето за смяна на личните данни избира да смени паролата си. Първо въвежда старата парола а след това новата и потвърждава с натискането на бутон за потвърждаване.

use case 15: Промяна на паролa

scope: N/A

level: user-goal

primary actor: лектор, администратор

stakeholders & interesets:

   - Лектор: Иска да промени паролата си и да въведе нова.

   - Администратор: Иска да промени паролата си и да въведе нова.

trigger:

   Потребителят е влязъл в системата и си влиза в профила.

preconditions:

   Лекторът/Администраторът трябва да е влязъл в системата

postconditions:

   Инициаторът е променил успешно паролата си.

main success scenario:

   01. Потребителят отива на профил страницата

   02. Системата предоставя личните данни, които има лектора

   03. Потребителят избира функционалността за смяна на паролата

   04. Системата предоставя полета за попълване на паролата

   05. Потребителят попълва полетата. за стара и нова парола

   06. Системата валидира новата парола

   07. Системата показва съобщение за успешно променена парола.

   extensions:

   04a. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени полетата с паролите.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 05 от главният сценарий

special requirements:

   - системата променя паролата на лектора/администратора за по-малко от 0.5 секунди.

frequency of occurrence:

   - Винаги, когато администратор иска да промени паролата си.

други:

   N/A

**8. Редактиране на лектор от администратора**

Администраторът има възможност да променя данните на лектори във факултета. Ако случайно е допусната някаква грешка да не се налага да пресъздава акаунта от начало.

use case 16: Редактиране на лектор от администратора

scope: N/A

level: user-goal

primary actor: администратор

stakeholders & interesets:

   - Администраторът: Иска да промени данните на лектор

trigger:

   Администраторът влиза на страницата, която показва всички лектори и натиска лектора, който иска да редактира

preconditions:

   Администраторът трябва да е влязъл в системата

   Администраторът трябва да има поне една лектор

postconditions:

   Инициаторът е променил успешно данните на лектора.

main success scenario:

    01. Потребителят отива на страницата, която показва всички лектори и натиска лектора, който иска да редактира

    02. Системата предоставя данните на лектора

    03. Потребителят избира функционалноста за промяна на данните на лектора.

    04. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

    05. Потребителят променя данните, които иска да промени.

    06. Системата проверява дали указаните данни са правилни

    07. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

   extensions:

    06а. Валидацията на въведените данни не е успешна

01. Системата предоставя възможност на потребителя да промени всяка една от данните на лектора.

02. Системата сигнализира кои полета са попълнени грешно

03. Продължаваме със стъпка 05 от главният сценарий

special requirements:

   - системата променя данните на лектор за по-малко от 1 секунда.

frequency of occurrence:

   - няколко пъти в годината

други:

   N/A

**9. Премахване на стая**

Администраторът има възможност да премахва дадена стая. В случай, че някаква стая се ремонтира или вече не съществува да бъде премахната от системата и да не може да се запазват занятия за нея.

use case 17: Премахване на стая

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

 - Администратор: Може да премахва стая, ако бъде разрушена или е в ремонт.

trigger:

   Администраторът отива на страница, на която са показани всички стаи и избира опция за изтриване на стая

preconditions:

   Потребителят е влязъл в системата като администратор

postconditions:

 - Изтриване на стая успешно

main success scenario:

  01. Потребителят отива на страницата, на която са показани всички стаи

  02. Системата показва всички стаи.

  03. Потребителят избира стая, която иска да изтрие.

  04. Системата пита за потвърждение за изтриването на стаята.

    05. Потребителят потвърждава изтриването на стаята.

    06. Системата изтрива стаята.

    07. Системата изпраща съобщение за успешно изтриване.

extensions:

  05а. Потребителя отказва изтриване на стая

    - Продължаваме със стъпка 04 от главния сценарий

special requirements:

   Системата изтрива стая за около 0.5 секунди.

frequency of occurrence:

   Много рядко

други:

 N/A

**10. Редактиране на стая**

Администраторът има възможност да редактира данните на дадена стая. В случай, че някаква стая вече има различни характеристики (има компютри или мултимедия за пример) да може да бъде променена в системата.

use case 19: Редактиране на стая

scope: N/A

level: user-goal

primary actor: администратор

stakeholders & interesets:

   - Администратор: Иска да промени данните на стая

trigger:

   Потребителят е на страницата, където са показани всички стаи, избира стая и натиска редактиране на стая

preconditions:

   Администраторът трябва да е влязъл в системата

postconditions:

   Инициаторът е променил успешно данните на стаята.

main success scenario:

   01. Потребителят отива на страницата, където са показани всички стаи

   02. Системата показва всички стаи

   03. Потребителят избира стая

   04. Системата показва детайлите на стаята.

   05. Потребителят избира да промени стая.

   06. Системата предоставя възможност на потребителя да промени всяка една от данните на стаята.

   07. Потребителя променя данните, които иска да промени.

   08. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

   extensions:

    N/A

special requirements:

   - системата променя детайлите на стая за по-малко от 1 секунда.

frequency of occurrence:

   - Няколко пъти в годината

други:

   N/A

**11. Премахване на специалност**

Администраторът има възможност да премахва специалности. В случай, че някаква специалност не се предлага вече в университета да бъде премахната от системата и да не се генерират програми за нея.

use case 18: Премахване на специалност

scope: N/A

level: user-goal

primary actor: администратор

stackeholders & interesets:

 - Администратор: Може да премахва специалност, ако бъде закрита.

trigger:

   Администраторът отива на страница, на която са показани всички специалности и избира опция за изтриване на специалност

preconditions:

 - Потребителят е влязъл в системата като администратор

postconditions:

 - Изтриване на специалност успешно

main success scenario:

  01. Потребителят отива на страницата, на която са показани всички специалности

  02. Системата показва всички специалности.

  03. Потребителят избира специалност, която иска да изтрие.

  04. Системата пита за потвърждение за изтриването на специалността

    05. Потребителят потвърждава изтриването на специалността

    06. Системата изтрива специалността

    07. Системата изпраща съобщение за успешно изтриване.

extensions:

  05а. Потребителя отказва изтриване на специалност

    - Продължаваме със стъпка 04 от главния сценарий

special requirements:

   Системата изтрива специалност за около 0.5 секунди.

frequency of occurrence:

   Един път на 10 години

други:

 N/A

**12. Редактиране на специалност**

Администраторът има възможност да редактира данните на специалностите. В случай на промяна на някакви данни да не се налага да се пресъздава специалността.

use case 20: Редактиране на специалност

scope: N/A

level: user-goal

primary actor: администратор

stakeholders & interesets:

   - Администратор: Иска да промени данните на специалност

trigger:

   Потребителят е на страницата, където са показани всички специалности, избира специалност и натиска редактиране на специалност

preconditions:

   Администраторът трябва да е влязъл в системата

postconditions:

   Инициаторът е променил успешно данните на специалността.

main success scenario:

   01. Потребителят отива на страницата, където са показани всички специалности.

   02. Системата показва всички специалности

   03. Потребителят избира специалност

   04. Системата показва детайлите на специалността.

   05. Потребителят избира функционалноста за промяна на специалност.

   06. Системата предоставя възможност на потребителя да промени всяка една от данните на специалността.

   07. Потребителят променя данните, които иска да промени.

   08. Системата проверява дали указаните данни са правилни (курсове, капацитет, предмети)

   09. Системата показва съобщение в зависимост дали данните са променени успешно или има проблем с тях

   extensions:

    N/A

special requirements:

   - системата променя детайлите на специалност за по-малко от 1 секунда.

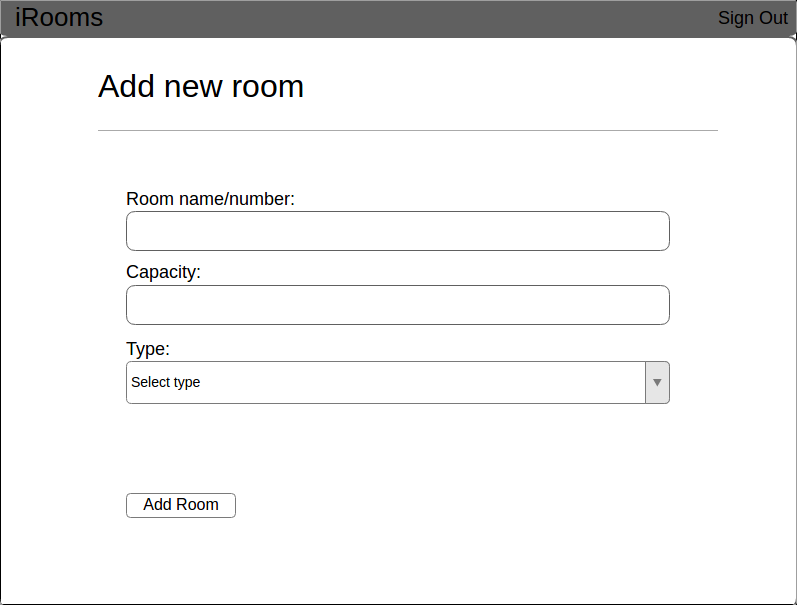
frequency of occurrence:

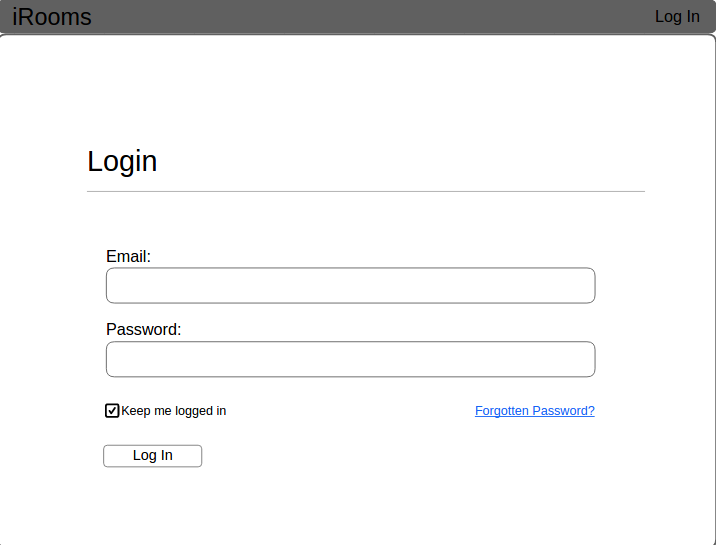
   - Една от важните функционалности за администратори

други:

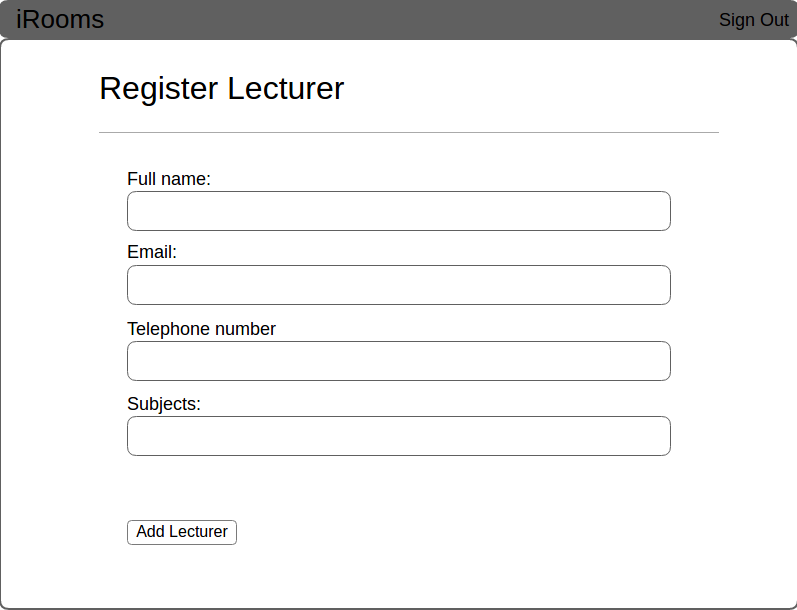
   N/A

## User interfaces









## Начално описание на нефункционалните изисквания по модела FURPS

## 

**1.Функционалност(Functionality)**

     1.1 Изпращане на нова парола - при забравена парола, системата изпраща нова,  произволно генерирана на имейла на потребителя.

     1.2 Възможност за допълнително описание на резервация - при резервация на зала, потребителят може да посочи допълнителна информация за това какъв ресурс ще съдържа залата (мултимедия, лаптоп и др.).

**2.Употреба (Usability)**

2.1   Администраторски акаунт – системата поддържа един администраторски акаунт.

2.2   Администраторски интерфейс – администраторът има специален интерфейс, чрез който вижда информация за системата и потребителите и ги администрира.

2.3   Лекторски интерфейс – лекторите имат собствен интерфейс, който им помага да рекуестват за резервация и да следят своите резервации.

2.4   Съвместимост с мобилни устройства – инетрфейсът на системата работи адекватно на различните мобилни устройства (телефони, таблети).

**3.Надеждност (Reliability)**

3.1   Криптиране на пароли – системата поддържа паролите в криптиран вид.

3.2   Устойчивост на данни – няма да се губят данни, въведени преди повече от 24 часа.

3.3   Downtime – системата няма да е в неизправност от повече от 48 часа сборно на година.

3.4   Online backup – системата запазва състоянието си веднъж на всеки 24 часа и продължава да е в изправност докато този процес се изпълнява.

3.5   „Живот“ на backup – можем да се върнем не повече от месец назад.

3.6   „Живот“ на лекторска сесия – лекторските сесии в системата изтичат автоматично при 1 час неактивност на потребителя.

**4.Производителност(Performance)**

4.1   Бързина на зареждане на календара – не повече от секунда и половина.

4.2   Брой посещения – поддържа около 100-150 посещения на ден.

4.3   Брой – активни акаунти – системата поддържа до 1000 активни лекторски акаунта.

**5. Поддръжка(Supportability)**

5.1   Поддръжка на Linux – системата е съвместима с Linux и дъщерните му дистрибуции.

5.2   Жизнен цикъл на системата – 10 години

**6. Ограничения по интерфейса (Interface constraints)**

6.1   Уеб интерфейс – интерфейс достъпен чрез стандартните уеб браузъри (Chrome, Mozilla, IE/Edge, Opera, Safari).

6.2   Мобилен интерфейс – интерфейс, който да функционира адекватно на мобилни устройства.

**7.Ограничения по дизайна (Design constraints)**

     7.1 База данни – съхранението на данни се осъществява чрез релационна база             данни.

**8.Ограничения по имплементацията (Implementation constraints)**

8.1   Linux платформа – всичко, което правим трябва да е съвместимо и легално в Linux.

## Речник

Част 1) Термини от предметната област:  
   
 \*) видео проектор – дигитално устройство, което визуализира снимки или видео материал върху сравнително голяма площ  
  
  
Част 2)  Термини със специфично значение в контекста на проекта:

\*) администратор – лице, което отговаря за инсталирането, конфигурирането, управлението, обновяването и поддръжката на определен софтуерен продукт  
   
 \*) профил – е визуално представяне на набор от предоставени лични данни и информация, свързани с конкретен компютърен или онлайн потребител.  
  
  
Част 3) IT термини:  
  
       \*) Интеграция – Обединяване(свързване) на две или повече информационни системи  
  
       \*) Фийчър – част от функционалността на даден софтуерен продукт  
   
       \*) логин – вписване в информационна системата  
   
       \*) акаунт - синоним на профил

## Време

|  |  |  |
| --- | --- | --- |
| Студент | Задача | Време |
| Иван Атанасов | 6. Вашата начална визия за бъдещия продукт - функционалност, ограничения, елементи на потребителски интерфейс; изграждане на модели | 4:00 часа |
| Ивелин Тодоров | Обобщение на записките на екипа под формата на описание на наблюденията ни при определяне изискванията; изграждане на модели | 3 часа |
| Богомил Богомилов | Методи за определяне на изискванията и разпределяне на хората от екипа; изграждане на модели | 3 часа |
| Александър Танков | Описване същността на осъществяваната от нас идея; изграждане на модели | 4 часа |
| Валентин Георгиев | Презентацията, изграждането на модели, разпределяне на задачите в екипа | 4:30 часа |
| Иво Райков | Изграждане на модели | 3 часа |
| Георги Димов | Изграждане на модели + презентация | 3:30 часа |

## 

## Време

|  |  |  |
| --- | --- | --- |
| Студент | Задача | Време |
| Иван Атанасов | Обсъждане на проекта с колеги, работа върху пропуските от минали път, работа върху изграждането на финалната версия на проекта; оформяне на документацията | 12 часа |
| Ивелин Тодоров | Изграждане на UML диаграми; събирания и обсъждане на проекта с колеги | 11 часа |
| Богомил Богомилов | Събирания и обсъждане на проекта с колеги; изграждане на UML диаграми | 11 часа |
| Александър Танков | Събирания и обсъждане на проекта с колеги, поправяне на потребителските случаи посредством критиките след презентацията, работене вътху domain model-а | 15 часа |
| Валентин Георгиев | Изграждане на UC\_brief, пълно описание на всички UC, както и приоритизирането им; направа на презентация | 20 часа |
| Иво Райков | Оформление на документа, преработка на CD модели | 10 часа |
| Георги Димов | Събирания и обсъждане на проекта с колеги,  Domain Model | 15 часа |