ПРОЕКТИРАНЕ НА ЧОВЕКО-МАШИНЕН ИНТЕРФЕЙС

Курсов проект

SIMS

СОФИЙСКИ УНИВЕРСИТЕТ

„СВ. КЛИМЕНТ ОХРИДСКИ“

основан 1888 г.

Стефан Велев, фак. №: **62537**
Даниел Халачев, фак. №: **62547**
Павел Атанасов, фак. №: **62555**
Давид Петров, фак. №: **62596**
спец. **Софтуерно инженерство**, **III** курс
Факултет по математика и информатика

Съдържание

1. Въведение. Цели и задачи на системата

В България има 52 акредитирани висши училища. Всяко едно от тях обучава хиляди студенти, за всеки от които се генерира и съхранява огромно множество от информация. За да бъдат тези данни използваеми, те трябва да бъдат съхранявани в подходящи структури и на подходящо място по единен начин.

За съжаление, твърде голяма част от съществуващите системи не решават изцяло този проблем, а в допълнение пораждат и други проблеми.

Нека разгледаме Системата за управление на студентската информация (СУСИ) на Софийския университет. Тя притежава множество функционалности, които други системи (например УИСС на Техническия университет) не поддържат - записване на избираеми дисциплини, преглед на оценки и кредити по дисциплини, заплащане на семестриална такса, попълване на оценки. Същевременно, подобно на голяма част от студентските информационни системи в България, тя все още не поддържа елементарни, но наистина необходими функционалности - преглед на седмичния график, заплащане на всички видове такси, изцяло електронно попълване на оценки и протоколи, съхраняване и категоризиране на цялата необходима информация за документиране на образователния процес.

В допълнение, нито една от разгледаните от нас информационни системи не спазва принципите на качествения потребителски интерфейс - достигане до функционалност с до 3 кликвания на мишката, акцент върху разпознаване, а не запомняне, интуитивно и смислено разположение на бутоните, и други. Затова не е изненада негативната обратна връзка относно потребителското изживяване - студенти и преподаватели често не харесват съществуващите системи (включително СУСИ), а някои посочват и съмнения в сигурността на използваната от тях система - мнения, които са основателни с оглед на използваните технологии (в случая със СУСИ - ASP.NET, 2003 година).

Предложената от нас система - SIMS (Student Information Management System), цели да разреши проблема с електронното управление на студентската информация с основен фокус върху Софийския университет. Вниманието е насочено към няколко основни насоки за развитие:

- подобряване на потребителското изживяване за студенти и преподаватели

- улесняване на дейността на студентите чрез предоставяне на необходимата за тях информация, включително седмичен график, изчисление на оставащите кредити, заплащане на всички видове такси и други
- улесняване на дейността на преподавателите, например чрез изцяло електронно попълване на оценки и протоколи без това да се отразява на сигурността, съхраняване и автоматично предоставяне на администраторите на необходимата информация при атестация и други форми на контрол на качеството.
- улесняване на дейността на администраторите на информация чрез по-голямо дигитализиране на административните услуги и намаляване на човекопотока на място.
- подобряване на сигурността чрез използване на съвременни технологии, чиято поддръжка не зависи от корпорации (за разлика от ASP.NET например).
- голяма част от недостатъците на съществуващите системи се дължат на факта, че всеки университет, въпреки ограничените си времеви, финансови и кадрови ресурси, разработва своя система, която впоследствие няма възможност да поддържа или подобрява самостоятелно. Затова като незадължителна посока на развитие очертаваме open-source разпространение на системата отвъд пределите на Софийския университет, което би позволило университетите да си сътрудничат, обединявайки усилията си в създаването и поддържането на една единствена, но наистина качествена система. Основание за тази идея дава сходството на данните, които университетите съхраняват - имена на дисциплини, студенти и преподаватели, оценки, факултетни номера, протоколи и други. Пример за успешно такова сътрудничество е системата Moodle, която вече няколко години се използва от множество университети по света и се харесва от повечето студенти и преподаватели. Наясно сме, че този подход ще създаде трудности като мигрирането на базите данни и интеграцията с вече съществуващи системи, но за тези проблеми съществуват решения - инструменти за database migration, подходящ дизайн на базата данни (например използването на Бойс-Код нормални форми), interface adapters, REST API и други.

2. Целева група. Потребители

Целевата на група на системата са студентите, преподавателите и администраторите от Софийския университет, както и потенциални потребители от други български университети. Потребителите на системата SIMS ще бъдат четири роли:

- студент - лице, което се обучава в университета. Може да достъпва различна информация като седмичен график, нанесени оценки към текущия момент, брой изпити, брой получени кредити до този момент, данни за лицето, данни за учебния план според годината, в която е приет, данни за здравно осигуряване, данни за водените дисциплини,

координати за връзка с преподаватели и административни лица. Освен това може да се записва за избираеми дисциплини, както и да заплаща такси (семестриални, за изпити и др.).

- преподавател - лице, което обучава студенти в университета. Може да прави справки за водените от него дисциплини, брой студенти, които посещават водения от преподавателя курс, съдържание на генерирани протоколи в края на семестъра. Също така може да нанася оценки в съответните протоколи, както и да комуникира със студентите чрез изпращането на съобщения до всеки един от тях.
- администратор на данни - лице, което въвежда информацията, която студентите и преподавателите достъпват (учебни дисциплини, графици, метаданни за профили, кампании за такси и избираеми дисциплини) и прави справки справки за студенти и преподаватели във връзка с учебната и икономическата активност на университета (проверка за платена такса при заверка на семестър, проверка на хорариум, проверка на записани студенти за избираема и други).
- администратор на системата - лице, което поддържа изправността на хардуера и софтуера на системата и отстранява неизправностите, възникнали при употребата ѝ. Може да прави промени в кода, базите данни, настройки на софтуера и хардуера, преглед на лог файлове.

3. Анализ на потребностите

Както стана ясно от горната точка, разпознаваме три категории потребители, респективно със следните извлечение от нас потребности:

За всички:

- системата ще подобри вече съществуващите функционалности в СУСИ (и евентуално други подобни системи) чрез по-строго придържане към правилата за интуитивен графичен интерфейс.

За студентите:

- организирането на седмичен график е трудоемко, ако трябва всеки студент да извлича информацията поотделно за себе си чрез посещение на място или гледане на физически разпечатани графици;
    - С нашата система студентите ще имат възможност да преглеждат официален график на занятията в дигитален формат.
- при еднократни промени в графика (предвидени или непредвидени), често е трудно новината да достигне до всички заинтересовани студенти навреме;
    - Затова нашата система предлага функционалност за масово изпращане на съобщения до всички участници в курс (от преподавателя за отложено занятие, например), както и непосредствено известяване (по имейл / СМС) за да може информацията да достигне навреме до всички.
- заплащането на семестриални такси или такси за поправителни изпити чрез физическо посещение до банка / каса на университета е времеемко и неудобно.
    - Чрез СИМС студентите ще могат да осъществят плащането директно по банков път чрез системата или да си генерират код за плащане на каса на EasyPay от удобно за тях място.

- студентската книжка все още съществува и е официално признат задължителен документ, но всички знаем колко време и усилия коства (не само на студентите) попълването ѝ от всички преподаватели всеки семестър. На практика информацията вече се съхранява предимно дигитално, и това е разбираемо - по-удобно и достъпно е, а и в почти всички случаи е по-сигурно. Затова СИМС предоставя на студентите възможността да преглеждат цялата си академична история в институцията до момента - (не/)завършени (избираеми/задължителни) курсове със съответните оценки, средни оценки по семестри, събрани кредити, оставащ хорариум и т.н. Всичко това в обединен прегледен изглед на страницата за общ преглед на дисциплини.
- записването на избираеми дисциплини е сложен за управление процес поради голямото разнообразие от курсове и различните разпределения на студенти в тях. Затова е реалистично немислимо да се случва на хартия в днешно време и е задължителна част от всяка система за управление на университетска информация.
    - Нашата система не е твърде революционна (за СУ) в предлагането на възможност за директно дигитално записване на избираеми дисциплини, но затова пък залага на надеждност и удобство. Всички знаем за проблемите с производителността на сегашната СУСИ, но освен отстраняването на този проблем, СИМС ще предоставя на студентите и възможност за маркиране на избираеми като фаворити, за да може, когато дойде време за записване, решението да се вземе на базата на по-прегледно филтрирани избрани дисциплини.
- сигурността е от ключова важност при работата с чувствителни лични данни. В повечето университетски системи е много трудно обикновения потребител да забележи проникване в профила си, преди да е твърде късно (да са настъпили осезаеми злоупотреби).
    - Затова, освен стандартното криптиране на данните, СИМС интегрира един първичен, но пък достатъчно ефикасен допълнителен метод за своевременна превенция на неоторизиран достъп - студентите ще могат да преглеждат непроменяем лог на последните минимум 20 влизания в системата, така че неверифицирана активност да може да бъде лесно забелязана.

За преподавателите:
Системата се фокусира върху двете основни според нашите проучвания дейности, които вълнуват преподавателите:

- съставянето на курс и без това е трудоемка работа - от самото му концептуализиране, през подготовката на конспект, до оформянето на формалностите покрай администрацията. СИМС цели да направи именно последното възможно най-лесно за преподавателите, като им предоставя удобна и компактна УЕБ форма за попълване на нужните административни данни по изготвянето на курса (като хорариум, препоръчителен семестър за слушане, метод на оценяване, предварителни знания и пр.) или ако вече е създаден след съгласуване с ръководството - за промяна на анотацията.
- въвеждането на обратна връзка в края на семестъра. Става дума, разбира се, за протоколите с официалните крайни оценки на студентите след приключване на курса. Системата ще позволява автоматичното генериране на протокол с

всички участници в курса, като преподавателят ще може да нанася оценките в него или една по една (примерно директно след поточно устно изпитване), или да импортира накуп от файл във формат .csv или .xsl/.xslx, след като (например) си е нанесъл сам оценките, както му е удобно.
- Разбира се, проблемът със спонтанното отменяне/отлагане на занятията и поддръжката на унифициран и публично достъпен от всички заинтересовани от курса лица график важи с пълна сила и от тази половина на учебния процес. Затова СИМС позволява на преподавателите да изпращат масови съобщения до участниците в курс.

За администрацията:

- първото нещо, което интересува ръководството на университета, е именно екипът - преподаватели и други администратори. Затова СИМС предоставя на администраторите на данни възможността да създават потребителски профили в системата за всеки един от тях, като така актуално действащите лица в университета могат да се проследят освен бюрократично, и дигитално, и да ползват услугите на системата.
- администраторите служат като първичния източник на истината, така че реално те оторизират най-съществените дейности - като например въвеждането на нов курс. След изготвянето му от преподавателя и одобрение от ръководните кадри, административен служител създава нова курс в системата и той започва да се води и да е наличен за студентите. Управлението на курсовата дейност изцяло на хартия е реликва от миналото и макар потенциално да е проблем, който системата решава, това се подразбира.
- абсолютно аналогично системата улеснява всички други дейности по създаване / обновяване и премахване на артефакти, като:
    - верифициране на финалните протоколи с оценки от преподавателите;
    - управление на участниците (записване / премахване на студенти) в курс;
    - обявяване и промени на кампании за избираеми дисциплини;

Общият проблем, който решава системата от гледна точка на администрацията, е управлението на огромни обеми от данни, което е непосилно без дигитализация. В този ред на мисли основният акцент и иновация идват в удобството за боравене и безопасността на данните. Най-често срещаното оплакване от изпробваните досега решения, което чухме, а и предимно от интервютата с административните кадри, беше колко е лесно да се наруши коректността на данните, и то незабелязано, докато не стане голям проблем. Най-силното желание у потребителите е невалидни данни да са (ако не невъзможни), то много трудни за въвеждане, системата да помага с допълнителна валидация и действията да са ясно обозначени и маркирани с еднозначни индикатори (съобщения, предупреждения, потвърждения на намерението).

Например, при действие “потвърждаване” на получен протокол от преподавател, системата ясно да предупреждава, че тези оценки ще станат видими за студентите като крайни и действието е необратимо.

4. Спецификация на изискванията

** Анализът на изискванията по отношение на тяхното приоритизиране е направен чрез прилагане на техниката **MoSCoW**.*

4.1 Функционални изисквания

1. Гост-потребител

1.1. Вписване:

1.1.1. Системата ще предоставя възможност за вписване в системата.

1.1.2. За вход системата ще изисква валидни потребителско име и парола.

1.1.2.1. Системата трябва да позволява имейл да бъде валидно потребителско име.

1.1.3. Ако липсва запис за потребител в базата данни, системата трябва да визуализира съобщение за грешка.

1.1.4. Системата ще предоставя възможност за възстановяване на забравена парола чрез изпращане на имейл за промяна на паролата към потребителя.

1.1.4.1. Имейлът за възстановяване на паролата би могъл да съдържа линк, препращащ към страница за промяна на паролата.

1.1.4.2. Имейлът за възстановяване на паролата би могъл да съдържа код, който системата ще очаква да бъде въведен, за да завърши промяната на паролата.

1.1.5. След успешен вход потребителят ще бъде препратен към началната страница на своя профил.

1.2. Преглед на новини:

1.2.1. Системата би могла да показва публична информация за предстоящи и протичащи в момента на ползване кампании за плащане на такси, записване на избираеми и други.

2. Студент и преподавател - общи

2.1. Преглед на седмично разписание:

2.1.1. Системата ще предоставя на студентите и преподавателите персонализиран ежеседмичен график на занятията, в които участват, под формата на таблица.

2.1.2. Системата ще предоставя информация единствено за дисциплините, записани от студента или водени от преподавателя.

2.1.3. Всяко занятие ще съдържа данните: име на дисциплина, място и час на провеждане и ще има етикет за метаданни - лекция/упражнение и задължителна/избираема.

2.1.4. Студентите и преподавателите ще могат да редактират личното си седмично разписание.

2.1.4.1. Редактирането ще позволява промяна на времето на провеждане.

2.1.4.2. Редактирането ще позволява скриване на занятието в разписанието.

2.1.4.3. Системата ще позволява връщане към разписанието по подразбиране.

2.2. Преглед на съобщения:

2.2.1. Студентите и преподавателите ще могат да преглеждат получените съобщения.

2.3. Изпращане на съобщения:

2.3.1. Потребителите ще могат да изпращат съобщения, въвеждайки:

2.3.1.1. адресат;

2.3.1.2. тема;

2.3.1.3. съдържание на съобщението.

2.3.2. Системата би могла да позволява прикачване на файлове към съдържанието на съобщенията.

2.3.3. Ако потребителят е преподавател, системата ще позволява адресиране на повече от един потребител - група, курс, специалност.

2.4. Системата би могла да предоставя възможност за настройка на имейл типа известия, които потребителят да получава спрямо ролята си

2.4.1. Студент: нова семестриална оценка, нова такса, успешно платена такса, ново съобщение, нова кампания, промяна на позиция в опашка за записване на избираема дисциплина, промяна в седмичното разписание.

2.4.2. Преподавател: ново съобщение, приет протокол, отхвърлен протокол, промяна в седмичното разписание.

2.5. Системата би могла да предоставя персонализирана начална страница с най-използваните дейности на потребителя.

2.5.1. Системата би могла да предоставя възможност за настройка на видимостта на дейностите преките пътища от началната страница.

2.6. Системата ще позволява на потребителите да виждат списък на последните 20 вписвания в профила си в системата.

2.6.1. Прегледът на вписванията ще включва информация за датата и часа, браузъра и типа на устройството.

2.6.2. Прегледът би могъл да включва информация за IP адреса на устройството, от което е направено вписването, или приблизителното му местоположение, изчислено с помощта на IP адреса.

2.7. Смяна на паролата:

2.7.1. Системата ще предоставя изглед за смяна на паролата.

2.7.2. Системата ще изисква двуфакторно удостоверяване преди въвеждането на новата парола.

2.8. Изпращане на молба:

2.8.1. Системата би могла да предоставя възможност за електронно попълване и изпращане на заявление до администрацията на университета.

2.8.2. Потребителят би могъл да избира измежду готов списък от бланки

2.8.3. Администраторите биха могли да бъдат лицата, които качват готовите бланки в системата.

2.8.3.1. Системата би могла да предоставя начин за създаване на готовите бланки от администратор на данни с цел да се избегне нуждата от HTML специалист за създаването им.

3. Студент

3.1. Преглед на избираемите дисциплини:

3.1.1. Системата ще предоставя на студент списък с всички налични за записване избираеми дисциплини.

3.1.1.1. Системата ще предоставя наличните за текущия семестър избираеми дисциплини, ако има активна кампания за избираеми дисциплини.

3.1.1.2. Системата ще предоставя наличните за следващия семестър избираеми дисциплини, ако няма активна кампания за избираеми дисциплини.

3.1.1.3. При липса на въведена информация за бъдещи избираеми дисциплини, системата ще визуализира избираемите дисциплини от съответния семестър на предишната година.

3.1.2. Списъкът ще включва името на дисциплината, описанието ѝ (ако е попълнено), изискванията за записване (ако има такива), броя кредити, които носи, и титуляра, от когото се води дисциплината.

3.1.3. Системата би могла да включва други метаданни за дисциплините в списъка с избираеми дисциплини.

3.2. Маркиране на избираеми дисциплини:

3.2.1. Системата ще позволява на студент маркиране (отбелязване) на избираеми дисциплини от списъка с налични избираеми дисциплини по време на преглед на избираемите дисциплини.

3.2.2. Системата ще позволява премахване на маркирането

3.2.3. Системата ще позволява неограничен брой повторни маркирания

3.2.4. Системата ще предоставя изглед за преглед на маркираните избираеми дисциплини

3.2.5. Списъкът с маркирани избираеми автоматично ще отстранява маркираните избираеми дисциплини, които са

записани окончателно, след края на кампанията за избираеми дисциплини.

3.3. Записване на избираеми дисциплини:

3.3.1. Системата ще позволява на студентите да запишат избираема дисциплина:

3.3.1.1. директно, ако има свободни места (при зададен лимит)

3.3.1.2. в опашка, ако няма свободни места (при зададен лимит)

3.3.1.3. само ако отговарят на критериите за записване

3.3.1.4. в момента е активна кампания за избираеми дисциплини

3.3.2. Системата ще позволява на студентите да филтрират списъка с избираемите дисциплини по следните критерии: име на дисциплината,титуляр, брой кредити.

3.3.3. Системата би могла да позволява филтриране и по други критерии.

3.3.4. Системата ще позволява на студентите да сортират избираемите дисциплини по следните критерии: име на дисциплината,титуляр, брой кредити.

3.3.5. Системата би могла да позволява сортиране и по други критерии.

3.3.6. Системата трябва автоматично да записва студент, който е първи в опашката, при освобождаване на свободно място в списъка за дадената дисциплина.

3.3.7. Системата би могла да не съхранява данните за опашката след края на кампанията за избираеми дисциплини.

3.3.8. Системата ще предоставя неограничен брой възможности за отписване на записаните дисциплини до края на текущия етап в кампанията за избираеми дисциплини.

3.3.9. Системата ще предоставя неограничен брой възможности за повторно записване на избираемата дисциплина, при отписване по време на един етап в кампанията за избираеми дисциплини.

3.3.10. Системата няма да позволява повторно записване на избираема дисциплина, ако тя е била записана и изкарана от студента.

3.4. Плащане на такси:

3.4.1. Системата ще позволява на потребителите да извършват плащания на всички такси чрез системата ePay.

3.4.2. Системата ще показва точния размер на таксата, както и основание за таксата.

3.4.3. Системата ще позволява заплащане на семестриални такси

3.4.4. Системата ще позволява заплащане на такси за повторно явяване на изпити.

3.4.5. Системата би могла да позволява заплащане на други такси.

3.4.6. Системата би могла да уведомява студентите за сроковете за плащане на семестриални такси:

3.4.6.1. два дни преди началото

3.4.6.2. два дни преди края

3.4.7. Системата би могла да скрива такси, които са заплатени или не могат да бъдат заплатени вече

3.5. Обобщен преглед на дисциплини:

3.5.1. Системата ще предостави на студентите възможност за преглед на получените оценки по дисциплини

3.5.2. Прегледът ще включва име на дисциплина, титуляр, етикет задължителна/избираема, семестър на полагане, оценка, кредити (ECTS).

3.5.3. Изгледът за преглед би могъл да предоставя и други метаданни.

3.5.4. Дисциплините в прегледа могат да се филтрират по:

3.5.4.1. Семестър (текущ / минали)

3.5.4.2. Задължителни / избираеми

3.5.4.3. Взети / невзети / взети на поправка

3.5.5. Изгледът би могъл да позволява допълнителни критерии за филтриране.

3.6. Преглед на здравно осигуряване:

3.6.1. Системата би могла да предоставя изглед за преглед на здравното осигуряване по месеци.

4. Преподаватели

4.1. Преподавател ще може да редактира анотацията на дисциплина, която води.

4.2. Попълване на протоколи:

4.2.1. Системата ще позволява на преподавателите да генерират протоколи в края на семестъра за всяка водена от тях дисциплина

4.2.2. Системата ще позволява да се вписват оценки във вече генериран протокол:

4.2.2.1. или индивидуални (за конкретен студент) директно от интерфейс на системата;

4.2.2.2. или масово импортирани от .csv, .xsl, или .xslx файл във вида “факултетен номер; оценка”

4.2.3. Системата ще прави резервно копие на въведените до момента оценки на всеки 5 попълвания на данни или на всеки 3 минути (по-малкото от двете).

4.2.4. Изгледът за редакция на оценки ще позволява редакция на оценката само за един студент в даден момент.

4.2.5. Системата ще позволява на преподавателя да отбелязва етикети за статуса на протокол - Незавършен, Завършен, Изпратен и Отхвърлен.

4.2.5.1. Етикетът Изпратен ще бъде автоматично задаван от системата при изпращане за валидация от администратор на данни.

4.2.5.2. Етикетът Отхвърлен ще бъде автоматично задаван от системата при отхвърляне на протокола от администратор на данни поради необходимост от корекции.

4.2.6. Системата ще позволява на преподавателя да изпраща протокола за верификация.

4.2.6.1. Системата няма да позволява редактиране на протокола след изпращането му.

4.2.6.2. Системата ще изисква двуфакторно удостоверяване преди изпращане на протокола.

5. Администратори на данни

5.1. Създаване на нови акаунти:

5.1.1. Системата ще предоставя на администраторите функционалност за създаване на нови акаунти за студенти и преподаватели в системата.

5.1.2. Системата ще изисква от администраторите да въведат необходимата информация за новия потребител, която включва задължително: име, фамилия, имейл адрес, номер на мобилен телефон, роля, факултетен номер (за студенти).

5.1.3. При регистриране на нов студент освен задължителната информация, ще се изисква задължително и ОКС, специалност, начин на финансиране.

5.1.4. При регистриране на нов преподавател освен задължителната информация, ще се изисква задължително и информация за неговата титла, както и катедра, от която ще бъде част.

5.1.5. Системата би могла да изисква и други данни, които са в съответствие с политиката на съответното висше училище.

5.1.6. Системата ще генерира парола по подразбиране.

5.1.7. Системата ще изпраща на новия потребител автоматично генерирано известие по email с информация за акаунта

5.1.8. Системата би могла да задължи потребителя да смени паролата си по подразбиране при първо влизане в системата.

5.2. Създаване на нови дисциплини:

5.2.1. Системата ще осигурява възможност за администратор на данни да създава нова дисциплина за дадена академична година.

5.2.2. Системата ще дава възможност за създаване на чисто нова дисциплина, за което се изисква посочване на: име, целева специалност, титуляр (преподавател), конспект, хорариум, задължителни/препоръчителни предварителни изисквания.

5.2.3. Системата ще дава възможност за създаване на нова дисциплина като копие на друга с възможност за промяна на данните.

5.3. Управление на кампании:

5.3.1. Системата ще позволява на администратор на данни създаването на кампании, сред които кампании за избираеми дисциплини и кампании за плащане на такси.

5.3.2. Системата ще позволява създаване на кампании за плащане на семестриални такси.

5.3.2.1. При създаването на кампанията, системата автоматично ще генерира такса за всеки студент.

5.3.3. Системата ще позволява създаване на кампания за избираеми дисциплини от администратор.

5.3.3.1. Системата ще позволява разделянето на етапи, при създаването на избираема дисциплина.

5.3.3.2. Системата ще позволява избор между следните правила за етапите на избираеми дисциплини:

5.3.3.2.1. Записване на избираеми

5.3.3.2.2. Отписване на избираеми

5.3.3.2.3. Отписване на избираеми от предходен етап

5.3.3.3. Правилата за отделните етапи ще могат да бъдат определяни от съответното висше училище.

5.3.3.4. Системата ще изисква въвеждане на списък с избираеми дисциплини, които са налични за конкретната кампания

5.3.4. Системата ще позволява редактиране на списъка с избираеми дисциплини преди и по време на кампания за избираеми дисциплини.

5.3.4.1. Системата ще позволява редактиране на началната дата на кампания за избираеми дисциплини преди настъпване на предишната поставена дата.

5.3.4.2. Системата ще позволява редактиране на етапи, ако не са започнали.

5.3.4.3. Системата ще позволява редактиране единствено на крайната дата на етапи, които са започнали.

5.4. Управление на участниците в дисциплините:

5.4.1. Преглед на списък с всички записани студенти за дадена дисциплина

5.4.2. Системата ще позволява премахване на студенти от списък на записани за дисциплина

5.5. Редактиране на дисциплина:

5.5.1. Системата ще позволява промяна на името, описанието, преподавателите и анотацията на дадена дисциплина

5.5.2. Системата ще позволява промяна на критерия за подреждане на записани студенти за дисциплина

5.5.2.1. Системата ще позволява избор между следните критерии за записване:

5.5.2.1.1. Лимит и подредба по последователност на записване

5.5.2.1.2. Лимит и подредба по среден успех

5.5.2.1.3. Без лимит (например при провеждане на входен тест)

5.5.3. Промяна на лимита за избираема дисциплина.

5.6. Верифициране на протоколи:

5.6.1. Системата ще позволява администратор на данни да верифицира протокол, изпратен от преподавател.

5.6.2. Системата ще позволява администратор на данни да отхвърля протокол, изпратен от преподавател за верификация.

5.6.3. Системата ще връща отхвърлен от администратора на данни протокол на преподавателя за редакция.

5.7. Създаване на справка:

5.7.1. Системата ще позволява на администраторите на данни да създават справки за студенти и преподаватели.

5.7.2. Системата ще позволява справките да се композират динамично спрямо ролята на лицето, за което се прави справката и данните за него в базата данни.

Пример: *избиране на роля -> ако е преподавател се появяват полета за избор на дисциплини и хорариум за тях; ако е студент - отново избор на дисциплини, но с избор на оценки и кредити за тях.*

6. Администратори на системата

6.1. Системата ще позволява администратор на системата да **създава профили** от тип **“администратор на данни”** чрез директни права за достъп до базата данни.

6.2. Системата ще позволява администратор на системата да **изтрива профили** от тип **“администратор на данни”** чрез директни права за достъп до базата данни.

4.2 Нефункционални изисквания

1. Сигурност:

1.1. Системата трябва да използва двуфакторно удостоверяване, когато еднофакторното не е достатъчно сигурно. .

1.2. Двуфакторното удостоверяване на системата трябва да се състои от парола и допълнителна стъпка - OTP код, изпратен по SMS или имейл.

1.3. Системата трябва да използва TLS протокол версия 1.2 или по-висока за всички комуникации.

1.4. Системата трябва да криптира всички лични данни на потребителите по време на съхранение.

1.5. Системата трябва да криптира всички лични данни на потребителите по време на предаването им по мрежата.

1.6. Максимален допустим коефициент на инциденти, застрашаващи данните, трябва да е 0,1%.

2. Мащабируемост:

2.1. Системата трябва да може да обслужва до 5 000 потребители едновременно без нарушаване на производителността.

3. Наличност:

3.1. Системата трябва да е налична в 99,5% от времето за месец.

3.2. Позволява се интервал за поддръжка от до един час веднъж месечно (по възможност в неработни за университета дни) между 23:00 и 06:00 часà.

4. Надеждност:

4.1. Системата трябва да има максимално допустим коефициент на грешки от 0,05% за всяка операция или транзакция.

5. Използваемост:

5.1. Дизайнът на потребителския интерфейс трябва да отговаря на правилата за добро потребителско изживяване:

5.1.1. Най-използваните 3 функционалности от всяка потребителска група трябва да бъдат достъпни с до 3 кликвания на мишката.

5.1.2. Изгледът за студенти и преподаватели трябва има десктоп и мобилна версия.

5.1.3. Потребителят не трябва да въвежда данни, които се съхраняват от системата в личния му профил и могат да бъдат изведени автоматично.

5.2. Системата трябва да предоставя уместни съобщения на потребителите:

5.2.1. Системата трябва да връща обратна връзка при невъзможно извършване на операция поради наложено ограничение от организацията.

5.2.2. Системата трябва да информира потребителя за настъпили грешки и изключения, като капсулира техническата информация и посочва единствено източник на грешката (локален компютър, мрежа или сървър) и възможност за разрешаване (презареждане на страницата, достъпване на системата в различно време, повторно вписване в системата).

6. Достъпност:

6.1. Системата трябва да бъде проектирана в съответствие със стандартите за достъпност на уеб съдържанието (WCAG) 2.1, ниво AA.

7. Съвместимост:

7.1. Системата трябва да позволява достъп до всички функционалности през следните браузъри: Firefox, Chromium, Chrome, Brave, Edge, Safari.

8. Производителност:

8.1. Системата трябва да има максимално време за отговор от 2 секунди за всяка операция или транзакция

8.2. Максимално средно време за зареждане на страница от 1 секунда.

9. Разработка на системата трябва да се осъществи чрез инструмент за контрол на версиите.

10. Леснота на поддръжка:

10.1. Системата трябва да има модуларен дизайн с коефициент на модуларност Relative System Modularity от 0.8.

10.2. Системата ще създава лог файлове, описващи всички сривове в системата.

10.3. Поддръжка на системата чрез система за контрол на версиите.

10.3.1. Препоръчителна система за контрол на версиите е git.

11. Цялост на данните:

11.1. Системата трябва да има поддържа периодични резервни копия на базата данни:

а) За последните три дни - на всеки два часа

б) За последните две седмици - последният от всеки ден

в) За последните три месеца - последният от всяка седмица

г) За последните две години - последният от всеки месец.

11.2. Механизми за възстановяване при бедствия, с максимален позволен риск от загуба на данни от 0,01%.

12. Защита на данните - GDPR:

12.1. Системата ще съхранява минималното достатъчно количество данни, необходимо за безпроблемната работа на всички функционалности на системата - пълна идентификация на

потребителя, извършване на учебната дейност и поне два метода за контакт.

12.2. Университетът, внедряващ системата, ще изисква писмена декларация за съгласие за ползване на системата преди предоставянето на данните за вход по подразбиране на всеки потребител.

12.2.1. Декларацията ще съдържа информация за данните, които системата събира за потребителите.

12.3. Университетът, внедряващ системата, ще въведе план за уведомление и действия, при изтичане на лични данни извън рамките на системата (с изключение на задължителните данни за плащане чрез външната система за плащане).

12.4. Системата ще съхранява личните данни в съответствие с GDPR и законите на Република България.

12.4.1. Системата ще съхранява личните данни на всички потребители през цялото време на тяхната активна длъжност в университета.

12.4.2. Системата ще съхранява личните данни на преподавателя 10 години след прекратяване на трудовата дейност в университета.

12.4.3. Системата ще съхранява личните данни на администратор на данни на системата 5 години след прекратяване на трудовата му дейност.

12.4.4. Университетът, който внедрява системата, се задължава да обнови сроковете на съхранение на личните данни на потребителите съгласно всякакви промени в GDPR или в законите на Република България.

5. Списък на потребителските случаи

UC-1 - Верифициране на протокол

Администратор на данни преглежда протокол, попълнен от преподавател, и потвърждава валидността му.
След верифициране, оценките в протокола стават видими за студентите.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	Преподавател
PRECONDITIONS	POSTCONDITIONS

Протокол е попълнен от преподавател и предаден за верифициране.	Системата показва нанесените в протокола оценки на студентите.

BASIC FLOW OF EVENTS

1. Потребителят отваря списък с получени протоколи за верифициране
2. Потребителят избира един протокол за верифициране от списъка
3. Потребителят приема протокола за коректен

ALTERNATIVE FLOWS

3.a Потребителят отхвърля протокола

1. Системата връща протокола за редактиране от преподавател

Related Requirements		
REQ-26	Верифициране на протоколи	Functional
REQ-26.1	Отхвърляне на протокол	Functional
REQ-26.1.1	Връщане на отхвърлен протокол	Functional

UC-2 - Редактиране на дисциплина

Администратор на данни редактира метаданните на дадена дисциплина.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	Администратор на данни

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е създал инстанция на учебната дисциплина.	Системата запазва промените в редактирането. От този момент всяка заявка за информация, включваща дисциплината, показва промените.

BASIC FLOW OF EVENTS

1. Потребителят избира дисциплина за редактиране
2. Потребителят променя метаданните на дисциплината
3. Потребителят прекратява редактирането
4. Потребителят запазва промените

ALTERNATIVE FLOWS

3.a Потребителят напуска страницата без да запази промените

1. Системата не запазва промените

Related Requirements		
REQ-25	Редактиране на дисциплина	Functional

UC-3 - Редактиране на кампания за избираеми дисциплини

Администратор на данни променя параметрите на вече създадена кампания за избираеми дисциплини.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни вече е създал кампания за избираеми дисциплини, която може да бъде редактирана.	Системата запазва промените за кампанията и те стават видими за всички участници в нея.

BASIC FLOW OF EVENTS

1. Потребителят избира кампания за избираеми дисциплини от списъка с вече създадени кампании.
2. Потребителят редактира параметрите на кампанията за избираеми дисциплини
3. Потребителят запазва промените

ALTERNATIVE FLOWS

2.a Потребителят не запазва промените

1. Системата не отразява и не запаметява промените

Related Requirements		
REQ-52.2	Редактиране на кампания за избираеми дисциплини	Functional
REQ-52.2.1	Редактиране на начало на кампания за избираеми	Functional
REQ-52.2.2	Редактиране на етапи	Functional
REQ-52.2.3	Редактиране на започнали етапи	Functional

UC-4 - Редактиране на кампания за семестриални такси

Администратор на данни редактира данни за кампания за плащане на семестриални такси.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
	Системата съхранява и отразява промените във всяка заявка за информация,

Администратор на данни вече е създал кампания за плащане на семестриални такси.	включваща кампанията за семестриална такса.

BASIC FLOW OF EVENTS
1. Администратор на данни избира създадена кампания за семестриални такси
2. Потребителят променя метаданните на кампанията
3. Потребителят запазва промените

ALTERNATIVE FLOWS
3.a Потребителят не запазва промените
    1. Системата не съхранява промените

Related Requirements		
REQ-52.1	Редактиране на кампания за семестриална такса	Functional

UC-5 - Създаване на дисциплина

Администратор на данни създава инстанция на дисциплина (само за конкретния семестър) и попълва метаданните ѝ.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
	Системата съхранява новата дисциплина и тя вече е видима за останалите потребители.

BASIC FLOW OF EVENTS
1. Потребителят отваря изгледа за създаване на дисциплина
2. Потребителят попълва задължителните данни и (евентуално) незадължителните данни
3. Потребителят запазва промените

ALTERNATIVE FLOWS
3.a Потребителят не запазва промените
    1. Системата не създава нова дисциплина

Related Requirements		
REQ-21	Създаване на дисциплина	Functional

UC-6 - Създаване на кампания за избираеми дисциплини

Администратор на данни създава кампания за избираеми дисциплини

PRIMARY ACTORS Администратор на данни	SUPPORTING ACTORS
PRECONDITIONS	POSTCONDITIONS Системата съхранява данните и създава нова кампания за избираеми дисциплини, която е видима за всички участници в нея.

BASIC FLOW OF EVENTS
1. Потребителят отваря изгледа за създаване на кампания за избираеми дисциплини
2. Потребителят въвежда задължителните и (евентуално) незадължителните данни за кампанията
3. Потребителят запазва промените

ALTERNATIVE FLOWS
3.a Потребителят не запазва промените
    1. Системата не създава нова кампания за избираеми дисциплини

Related Requirements		
REQ-22.2	Създаване на кампания за избираеми дисциплини	Functional
REQ-22.2.2	Въвеждане на списък с избираеми при създаване на кампания за избираеми дисциплини	Functional

UC-7 - Създаване на кампания за семестриални такси

Администратор на данни създава кампания за плащане на семестриални такси.

PRIMARY ACTORS Администратор на данни	SUPPORTING ACTORS
PRECONDITIONS	POSTCONDITIONS Системата създава нова кампания за семестриална такса, която става видима за всички участници. Системата създава нова такса за всички участници в кампанията автоматично.

BASIC FLOW OF EVENTS
1. Потребителят отваря изгледа за създаване на кампания за семестриална такса
2. Потребителят въвежда задължителните и (евентуално) незадължителните

данни за кампанията
3. Потребителят запазва промените

ALTERNATIVE FLOWS
3.а Потребителят не запазва промените
1. Системата не създава нова кампания за семестриална такса

Related Requirements		
REQ-22.1	Създаване на кампания за плащане на семестриални такси	Functional

UC-8 - Създаване на профил на преподавател

Администратор на данни създава профил на преподавател с парола по подразбиране и предоставя данните по подразбиране за вход в системата на съответното лице.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
	Системата съхранява данните и създава нов потребител/потребители

BASIC FLOW OF EVENTS
1. Потребителят избира опцията за създаване на един профил
2. Потребителят въвежда данните за профила
3. Потребителят запазва данните

ALTERNATIVE FLOWS
1.а Потребителят избира опцията за създаване на много профили
1. Потребителят импортира CSV файл, съдържащ данните на преподавателите
2. Системата запазва данните
3.а Потребителят не запазва данните
1. Системата не съхранява данните и не създава нов потребител

Related Requirements		
REQ-20	Създаване на акаунт на студент или преподавател	Functional
REQ-20.1	Информация при създаване на акаунт	Functional
REQ-20.1.2	Информация за преподавател при създаване на акаунт	Functional
REQ-20.1.3	Допълнителна информация при създаване на акаунт	Functional
REQ-20.3	Имейл при създаване на акаунт	Functional

UC-9 - Създаване на профил на студент

Администратор на данни създава профил на студент в системата с данни за вход по подразбиране. Възможно е това да става автоматично чрез импортиране на CSV файл с личните данни на студента, необходими за профила.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
	Системата съхранява данните и създава нови студенти.

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за създаване на един профил
2. Потребителят въвежда данните за профила
3. Потребителят запазва данните

ALTERNATIVE FLOWS

1.a Потребителят избира опцията за създаване на много профили
    1. Потребителят импортира CSV файл, съдържащ данните на преподавателите
    2. Системата запазва данните

3.a Потребителят не запазва данните
    1. Системата не съхранява данните и не създава нов потребител

Related Requirements		
REQ-20	Създаване на акаунт на студент или преподавател	Functional
REQ-20.1	Информация при създаване на акаунт	Functional
REQ-20.1.1	Информация за студент при създаване на акаунт	Functional
REQ-20.1.3	Допълнителна информация при създаване на акаунт	Functional
REQ-20.3	Имейл при създаване на акаунт	Functional

UC-10 - Създаване на справки

Администратор на данни избира филтри и критерии, по които системата да изведе необходима информация за студент или преподавател с цел заверка на семестър, информация за атестация и други.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
Търсеният Студент или Преподавател съществува и потребителят е влязъл като Администратор на данни.	Системата зарежда справка според избраните критерии и позволява експортирането ѝ в Word, Excel или PDF формат

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за създаване на справка
2. Потребителят избира опцията студент или преподавател
3. Системата показва контекстни филтри спрямо избраната роля
4. Потребителят въвежда информацията според филтрите
5. Потребителят започва генерирането на справката

ALTERNATIVE FLOWS

5.а Потребителят не започва генерирането на справката

1. Системата не извършва никакво действие

Related Requirements		
REQ-27	Създаване на справка	Functional

UC-11 - Създаване на профил на администратор на данни

Администратор на системата създава профил на Администратор на данни с парола по подразбиране.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на системата	

PRECONDITIONS	POSTCONDITIONS
	В системата успешно се създава профил на администратор на данни.

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за създаване на профил на Администратор на данни
2. Потребителят въвежда данните
3. Потребителят запазва данните

Related Requirements		
REQ-28	Създаване на профил на администратор на данни	Functional

UC-12 - Изпращане на съобщение

Преподавател създава ново съобщение до студент/студенти, което може да съдържа текстово съдържание.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател, Студент	

PRECONDITIONS	POSTCONDITIONS
В системата съществува желаният потребител, на когото да бъде изпратено съобщението	Системата изпраща съобщението на потребителя, който е адресат и му предоставя възможността да прочете съобщението.

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за изпращане на съобщение
2. Потребителят попълва темата на съобщение
3. Потребителят попълва адресат на съобщението
4. Потребителят попълва съдържанието на съобщението
5. Потребителят изпраща съобщението

ALTERNATIVE FLOWS

2.a Потребителят прикачва файл в съдържанието на съобщението

1. Системата прикрепя и доставя файла, заедно с текстовото съдържание

3.a Ако потребителят е преподавател, системата предоставя възможност за адресиране на група, курс и специалност

1. Системата изпраща съобщението до всеки един потребител в категорията на адресата

Related Requirements		
REQ-6	Изпращане на съобщения	Functional
REQ-6.1	Файлове в съдържанието на съобщенията	Functional
REQ-6.2	Адресиране на множество потребители в съобщение	Functional

UC-13 - Попълване на протокол

Преподавател попълва протокол с оценки, които са резултати от изпитна сесия. След това изпраща протокола за верифициране от администратор на данни. При потвърждение, оценките стават видими за студентите.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател	Администратор на данни

PRECONDITIONS	POSTCONDITIONS
Преподавателят е титуляр на поне една дисциплина.	Системата съхранява протокола. Ако протоколът е изпратен за верификация,

	администратор на данни получава протокола за верифициране.

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за попълване на протокол
2. Потребителят избира дисциплина, по която преподава
3. Системата предоставя изглед с възможност за нанасяне на оценката на всеки един студент, който участва в дисциплината
4. Потребителят нанася всички оценки
5. Потребителят запазва промените
6. Потребителят изпраща протокола за верифициране
7. Системата изпраща код за двуфакторно удостоверяване
8. Потребителят въвежда правилен код
9. Системата изпраща протокола за верифициране

ALTERNATIVE FLOWS

4.a Потребителят нанася част от оценките
    1. Потребителят запазва данните
    2. Потребителят нанася останалите оценки по друго време

5.a Потребителят не запазва промените
    1. Системата не съхранява данните и въведените оценки се губят.

8.a Потребителят въвежда грешен код
    1. Системата съхранява протокола, но не го изпраща за верифициране.

Related Requirements		
REQ-19	Попълване на протоколи	Functional
REQ-19.2	Резервно копие при попълване на протокол	Functional
REQ-19.4	Етикети на протокол	Functional
REQ-19.4.1	Етикет "Изпратен" на протокол	Functional
REQ-19.4.2	Етикет "Отхърлен" на протокол	Functional
REQ-19.5	Изпращане за верификация	Functional
REQ-19.5.1	Без право на редакция след изпращане на протокола	Functional
REQ-19.5.2	Двуфакторно удостоверяне преди изпращане на протокол	Functional

UC-14 - Записване на избираема дисциплина

Студент отваря списъка с избираеми дисциплини и маркира определен брой като записани.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е създал кампания за избираеми дисциплини.	Системата "записва" студента за избираемата дисциплина, ако отговаря на всички условия.

Кампанията за избираеми дисциплини е започнала. Студентът отговаря на критериите за записване на дисциплините, които е избрал.	Системата предоставя опцията за отписване от избираема след успешно записване.

BASIC FLOW OF EVENTS

1. Студентът избира опцията за записване на избираема дисциплина
2. Системата визуализира списък с всички избираеми дисциплини за семестъра
3. Студентът избира чрез графичния интерфейс опцията да запише избираема

Related Requirements		
REQ-14	Записване на избираеми дисциплини	Functional
REQ-14.1	Филтриране на списъка за записване на избираеми дисциплини	Functional
REQ-14.1.1	Допълнителни критерии за филтриране на избираеми за записване	Functional
REQ-14.2	Сортиране на списъка за записване на избираеми дисциплини	Functional
REQ-14.2.1	Допълнителни критерии за сортиране на избираеми дисциплини за записване	Functional
REQ-14.3	Автоматично преместване от опашката при записване на избираеми	Functional
REQ-14.6	Възможности за записване на избираема дисциплина	Functional
REQ-14.7	Повторно записване на изкарана избираема	Functional

UC-15 - Плащане на семестриална такса

Студент заплаща семестриалната си такса чрез външна система.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е започнал кампания за семестриални такси.	Системата съхранява данните за плащането и скрива таксата от изгледа на потребителя, който е заплатил таксата.

BASIC FLOW OF EVENTS

1. Студент избира таксата за заплащане
2. Системата препраща към външната система за плащане
3. Потребителят извършва плащането чрез външната система
4. Външната система регистрира плащането и информира студентската система

ALTERNATIVE FLOWS

3.а Потребителят не извършва плащането

1. Външната система не информира студентската система за плащането
2. Таксата остава неплатена

Related Requirements		
REQ-15.2	Плащане на семестриални такси	Functional

UC-16 - Плащане на такса за повторно явяване на изпит

Студент заплаща генерирана от системата такса за повторно явяване на изпит.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Студент неуспешно се е явил на изпит.	Системата съхранява данните за плащането и скрива таксата от изгледа на потребителя, който е заплатил таксата.

BASIC FLOW OF EVENTS

1. Потребителят избира предмет за повторно явяване
2. Потребителят избира опцията за генериране на такса
3. Потребителят избира таксата
4. Системата препраща към външната система за плащане
5. Потребителят извършва плащането чрез външната система
6. Външната система регистрира плащането и информира студентската система

ALTERNATIVE FLOWS

5.a Потребителят не извършва плащането

1. Външната система не информира студентската система за плащането
2. Таксата остава неплатена

Related Requirements		
REQ-15.3	Плащане на такси за повторно явяване на изпит	Functional

UC-17 - Преглед на здравно осигуряване

Студент преглежда здравното си осигуряване по месеци.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател, Студент	

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за преглед на здравното си осигуряване
2. Системата предоставя изглед, в който е видимо здравното осигуряване по

месеци.

Related Requirements		
REQ-17	Преглед на здравно осигуряване	Functional

UC-18 - Обобщен преглед на дисциплини

Студент преглежда данните за вече изкарани дисциплини - оценки, кредити, преподаватели, семестър на провеждане и др.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
За коректността на информацията е необходимо администратор на данни да е въвел данните за изкараните до момента дисциплини.	

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за преглед
2. Системата визуализира изглед с всички изкарани до момента задължителни и избираеми дисциплини с възможност за сортиране и филтриране

Related Requirements		
REQ-16	Обобщен преглед на дисциплини	Functional

UC-19 - Преглед на седмичното разписание

Студент или Преподавател преглежда седмичния график, който показва единствено неговата заетост по дни и часове в седмицата.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател, Студент	Администратор на данни

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е въвел в системата седмичния график.	

BASIC FLOW OF EVENTS

1. Потребителят избира опцията за седмичния график
2. Системата визуализира седмичния график на потребителя, без да включва дисциплини, които не посещава/на които не преподава.

Related Requirements		
REQ-4	Преглед на седмично разписание	Functional
REQ-4.1	Дисциплини в седмичното разписание	Functional
REQ-4.2	Информация в седмичното разписание	Functional
REQ-4.3	Редактиране на седмичното разписание	Functional
REQ-4.3.1	Редактиране на времето в разписанието	Functional
REQ-4.3.2	Скриване на занятие в разписанието	Functional
REQ-4.3.3	Връщане на разписанието по подразбиране	Functional

UC-20 - Преглед на събраните кредити

Студент преглежда изглед, в който е видимо колко кредити е събрал до момента.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни коректно е въвел броя кредити, които всяка дисциплина предоставя.	Списък с изкарани дисциплини за семестър/семестри, придружени с метаданни към тях, като оценка, изкарани кредити и др. Успешно е визуализиран списък с търсената информация от студента.

BASIC FLOW OF EVENTS

1. Студент отваря изглед за обобщен преглед на дисциплини.
2. Студент избира семестъра, за който иска да направи справката.
3. Системата визуализира съответния списък със записани дисциплини, каото всяка една от тях е придружена с данни като име, титуляр, семестър на полагане, оценка, кредити (ECTS) и др.

ALTERNATIVE FLOWS

2.а Студентът избира всички семестри преди генериране на заявката. След това се преминава се на стъпка 3.

UC-21 - Преглед на избираеми дисциплини

Студент преглежда списъка с всички избираеми дисциплини за дадения семестър.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
	Списъкът с избираеми дисциплини е прегледан от студента.

Администратор на данни е въвел всички възможни избираеми дисциплини в системата.

BASIC FLOW OF EVENTS

1. Студент отваря изглед за преглед на избираеми дисциплини.
2. Студент може да преглежда избираеми дисциплини в табличен вид.

Related Requirements		
REQ-12	Преглед на избираеми дисциплини	Functional

UC-22 - Маркиране на избираема дисциплина

От списъка с избираеми дисциплини студент има възможност да отбележи като маркирани предпочитани от него избираеми дисциплини. Маркирането на избираеми дисциплини не е обвързано по никакъв начин със записването им. То позволява при стартиране на кампанията за избираеми дисциплини да бъдат с приоритет за него най-горе в списъка.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е въвел всички възможни избираеми дисциплини в системата.	Маркираните избираеми дисциплини са най-горе в списъка на студента автоматично. При стартиране на кампания това позволява бързото им записване.

BASIC FLOW OF EVENTS

1. Студент отваря изглед за преглед на избираеми дисциплини.
2. Студентът отметва поле за маркиране срещу всяка избираема дисциплина, която желае да добави в списък на маркираните (запазването на избраните се осъществява автоматично от системата).

UC-23 - Преглед на лог файлове

Администратор на данни може да преглежда информация, която се е случила и запазила под формата на лог файлове. Там ще има информация за всички случили се сривове.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на системата	

PRECONDITIONS	POSTCONDITIONS

	Лог файловете са прегледани от администратора на системата.

BASIC FLOW OF EVENTS

1. Администратор на системата отваря изглед за преглед на лог файлове.
2. Администратор на системата избира лог файла, който желае да прегледа.

UC-24 - Вписване в системата

Гост потребител въвежда данните си за вход и влиза в системата, при което е пренасочен към началната страница спрямо ролята си.

PRIMARY ACTORS	SUPPORTING ACTORS
Гост потребител	

PRECONDITIONS	POSTCONDITIONS
Потребителят вече е бил регистриран от Администратор на данни или Администратор на системата (спрямо ролята си).	Системата пренасочва потребителя към началната му страница

BASIC FLOW OF EVENTS

1. Потребител навигира до ресурс от системата
2. Системата препраща Потребител до изглед за автентикация
3. Потребител въвежда данните си за вход (потребителско име/имейл и парола)
4. Системата верифицира коректността на данните
5. Системата пренасочва потребителя към началната му страница

ALTERNATIVE FLOWS

2.a Потребителят избира опцията за забравена парола

1. Потребителят е приканен от системата да въведе своя имейл адрес, с който е регистриран. На него той получава имейл с линк, препращащ към страница за промяна на паролата.

3.a Системата не намира потребител

1. На потребителя се показва уведомление, че данните за вход са некоректни, и той остава на същата страница.

Related Requirements		
REQ-1	Вписване в системата	Functional
REQ-1.1	Успешно вписване	Functional
REQ-1.2	Данни за вписване	Functional
REQ-1.3	Неуспешно вписване	Functional

UC-25 - Преглед на новини

Гост потребител преглежда информация под формата на новини в страницата за вписване в системата.

PRIMARY ACTORS Гост потребител	SUPPORTING ACTORS
PRECONDITIONS	POSTCONDITIONS Гост-потребителят има достъп до списък с актуални новини, свързани с дейността на университета.

BASIC FLOW OF EVENTS

1. Потребителят отваря началната страница за вписване в системата.

Related Requirements		
REQ-3	Преглед на новини	Functional

UC-26 - Преглед на вписванията

Системата ще позволява на студент или преподавател да вижда последните си 20 вписвания в системата, включително дата и час, браузъра и типа на устройството.

PRIMARY ACTORS Преподавател, Студент	SUPPORTING ACTORS
PRECONDITIONS Потребителят се е вписал успешно в системата.	POSTCONDITIONS Потребителят вижда кога е влизал за последно в системата и откъде.

BASIC FLOW OF EVENTS

1. Потребителят се вписва в системата
2. Потребителят отваря изгледа за преглед на вписванията (например в страницата за настройки и лични данни)

UC-27 - Промяна на парола

Вписан потребител прави заявка за смяна на паролата си за вписване в системата. Това се случва след двустъпкова автентикация, т.е. чрез въвеждане на текуща парола и OTP код. Новата парола се записва в базата данни.

PRIMARY ACTORS Администратор на данни, Преподавател, Студент	SUPPORTING ACTORS
PRECONDITIONS	POSTCONDITIONS

Потребителят се е вписал успешно в системата.	Паролата е сменена успешно, при коректни действия от потребителя. Паролата остава същата при некоректни действия на потребителя.

BASIC FLOW OF EVENTS

1. Потребителят се вписва в системата
2. Потребителят избира изгледа за смяна на паролата
3. Потребителят избира начин за двуфакторно удостоверяване
4. Системата изпраща код за двуфакторно удостоверяване по избраия начин
5. Потребителят въвежда получения код правилно
6. Системата предоставя изглед за въвеждане на старата и новата парола
7. Потребителят потвърждава смяната
8. Системата извежда съобщение за успешна смяна на паролата
9. Системата автоматично отписва потребителя от системата
10. Системата препраща потребителя на страницата за вписване

ALTERNATIVE FLOWS

5.a Потребителят въвежда получения код грешно
    1. Системата отказва промяна на паролата
6.a Потребителят въвежда грешно старата си парола
    1. Системата отказва промяна на паролата
6.b Потребителят въвежда грешно повторно новата си парола
    1. Система не позволява преминаване към промяна на паролата

UC-28 - Отписване на избираема дисциплина

Студент се отписва от избираема, която е записал.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Студентът е записал избираема дисциплина. Все още тече етап на кампания за избираеми дисциплини, който позволява отписване на дисциплината.	Студентът е отписан от избираемата дисциплина. Студентите след него се преместват едно място напред в списъка и опашката (ако има такава).

BASIC FLOW OF EVENTS

1. Потребителят навигира до изгледа за записани избираеми дисциплини
2. Потребителят намира записаната от него дисциплина
3. Потребителят отписва дисциплината чрез графичния интерфейс

Related Requirements		
REQ-14.5	Възможности за отписване от записана избираема дисциплина	Functional

UC-29 - Размаркиране на избираема дисциплина

Студент размаркира маркирана от него избираема дисциплина.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Студентът вече е маркирал поне една избираема дисциплина.	Системата премахва размаркираната дисциплина от списъка с маркирани дисциплини.

BASIC FLOW OF EVENTS

1. Потребителят отваря изгледа за маркирани избираеми дисциплини
2. Потребителят посочва желаната от него дисциплина за размаркиране

Related Requirements		
REQ-13.1	Размаркиране на избираема дисциплина	Functional

UC-30 - Редактиране на анотация на дисциплина

Преподавател редактира анотацията на водена от него дисциплина.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател	Администратор на данни

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е създал дисциплина и е посочил преподавател, който я води.	Системата запазва направените промени и ги отразява във всяка заявка за информация, свързана с дисциплината.

BASIC FLOW OF EVENTS

1. Потребителят избира водена от него дисциплина
2. Потребителят редактира анотацията на дисциплината
3. Потребителят запазва промените

ALTERNATIVE FLOWS

3.a Потребителят не запазва промените

1. Системата не отразява промените

Related Requirements		
REQ-18	Редактиране на анотация на дисциплина	Functional

UC-31 - Редактиране на кампания

Администратор на данни редактира метаданните на вече създадена кампания.

PRIMARY ACTORS	SUPPORTING ACTORS
Администратор на данни	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни вече е създал кампания.	Системата съхранява и отразява направените промени.

BASIC FLOW OF EVENTS

1. Потребителят избира вече създадена кампания с цел да я редактира
2. Потребителят редактира метаданните на кампания
3. Потребителят запазва промените

ALTERNATIVE FLOWS

3.a Потребителят не запазва промените

1. Системата не съхранява промените.

Related Requirements		
REQ-52	Редактиране на кампания	Functional

UC-32 - Редактиране на протокол

Преподавател редактира протокол, който е започнал, но не е довършил, или пък е бил върнат за редактиране след неуспешна верификация.

PRIMARY ACTORS	SUPPORTING ACTORS
Преподавател	Администратор на данни

PRECONDITIONS	POSTCONDITIONS
Преподавателят вече е създал протокол, който не е изпратил за верифициране, или е изпратил, но не е бил верифициран успешно и е върнат за редакция.	Системата съхранява промените, направени в протокола.

BASIC FLOW OF EVENTS

1. Потребителят избира протокол от списъка със съхранени протоколи
2. Потребителят нанася необходимите промени
3. Потребителят запазва промените

ALTERNATIVE FLOWS

3.a Потребителят не запазва промените

1. Системата не съхранява промените

UC-33 - Плащане на такса

Студент заплаща такса чрез външна система.

PRIMARY ACTORS	SUPPORTING ACTORS
Студент	

PRECONDITIONS	POSTCONDITIONS
Администратор на данни е започнал кампания за такси.	Системата съхранява данните за плащането и скрива таксата от изгледа на потребителя, който е заплатил таксата.

BASIC FLOW OF EVENTS

1. Потребитялт избира такса за заплащане
2. Системата препраща към външната система за плащане
3. Потребителят извършва плащането чрез външната система
4. Външната система регистрира плащането и информира студентската система

ALTERNATIVE FLOWS

3.a Потребителят не извършва плащането

1. Външната система не информира студентската система за плащането
2. Таксата остава неплатена

Related Requirements		
REQ-15	Плащане на такси	Functional

UC-34 - Изтриване на профил на администратор на данни

Администратор на системата изтрива профил на Администратор на данни.

PRECONDITIONS	POSTCONDITIONS
	Системата премахва профила и не позволява използването ѝ с него.

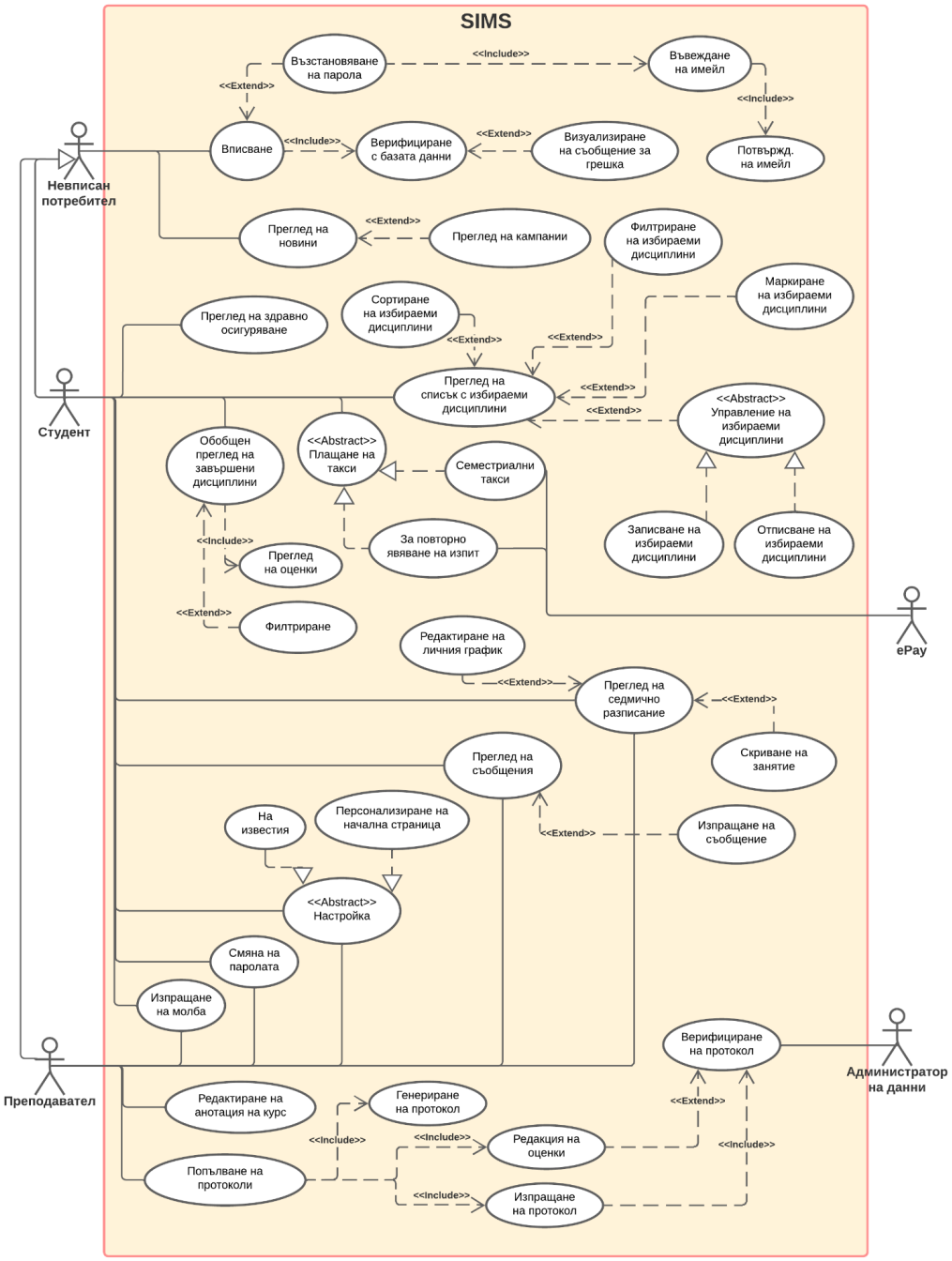
BASIC FLOW OF EVENTS

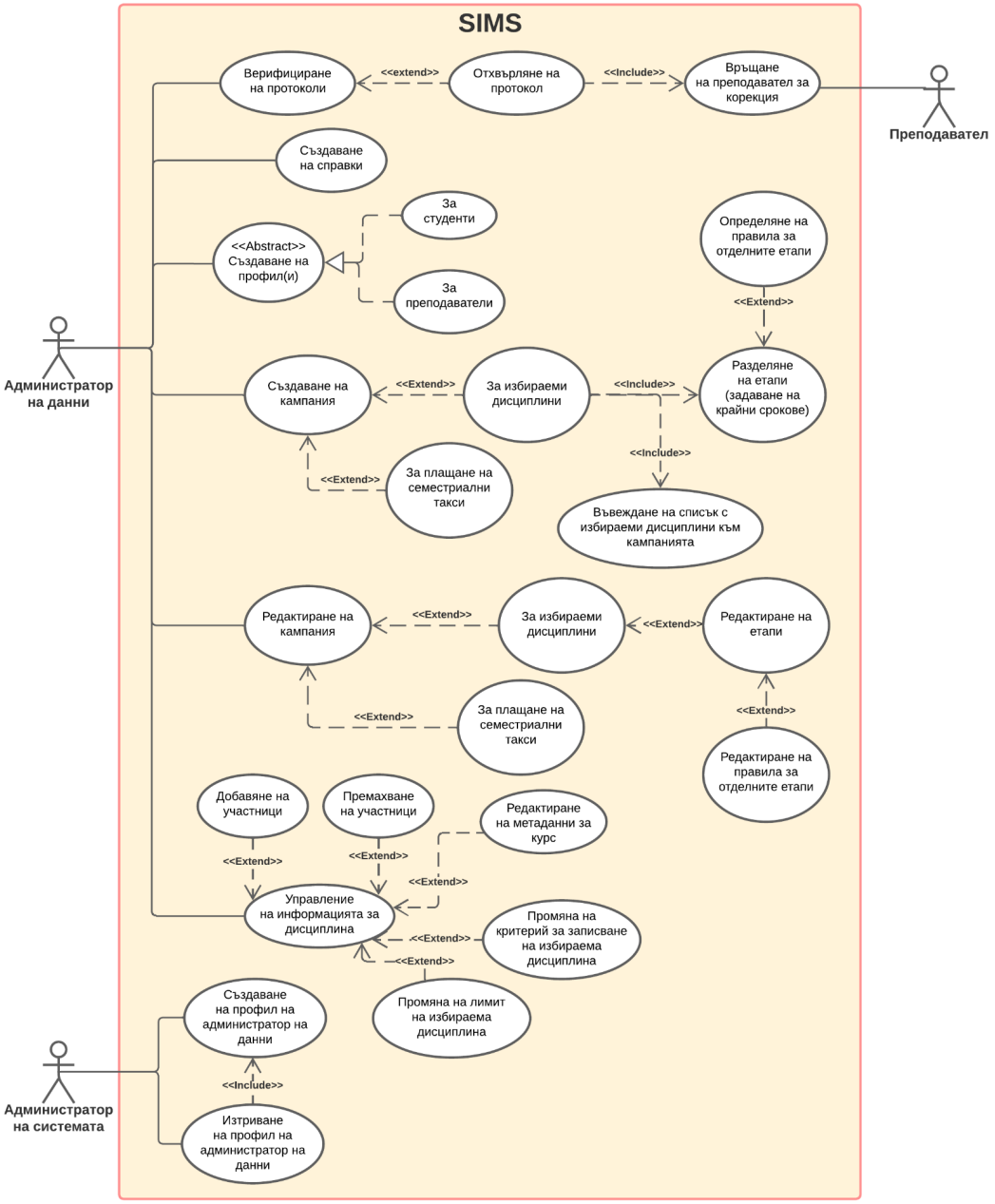
1. Потребителят избира профил на Администратор на данни
2. Потребителят потвърждава изтриването

Related Requirements		
REQ-29	Изтриване на профил на администратор на данни	Functional

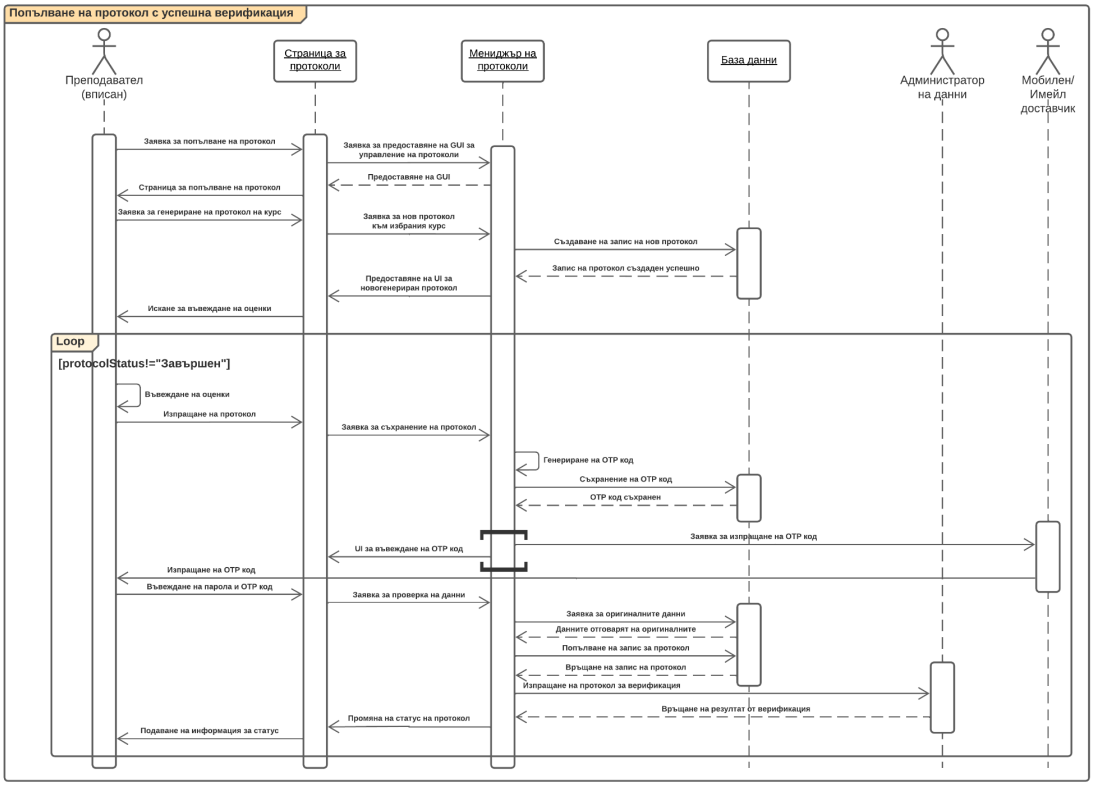
6. Аналитични модели на системата

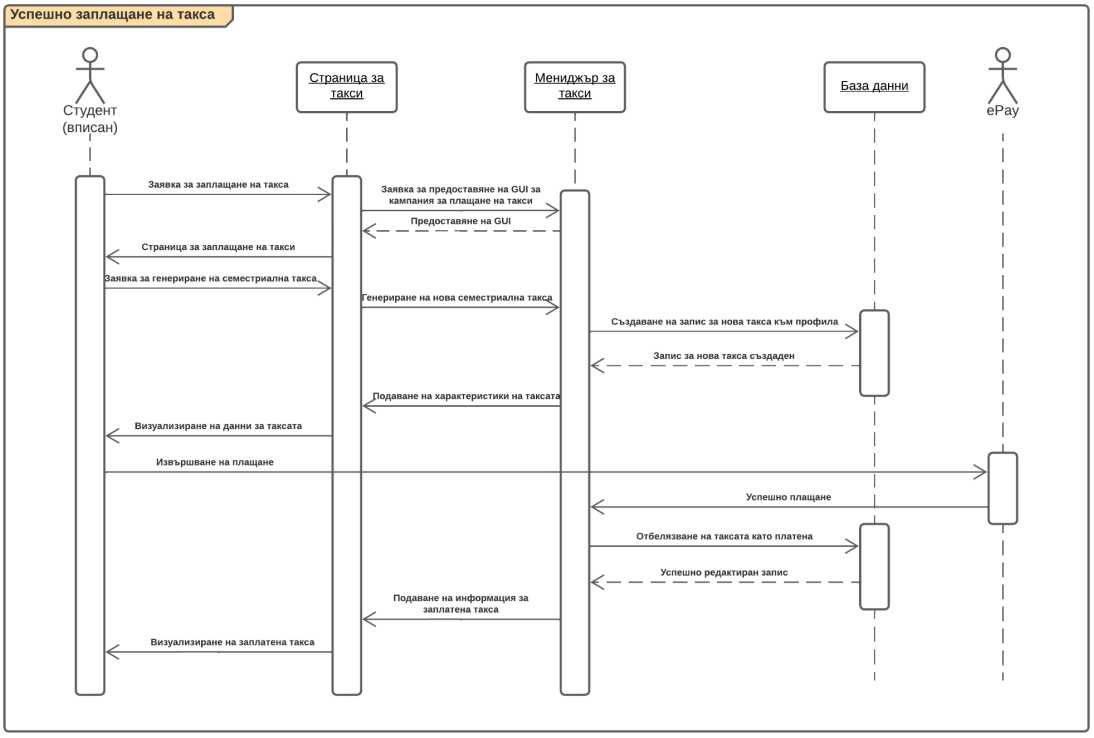
6.1 Диаграми на потребителските случаи



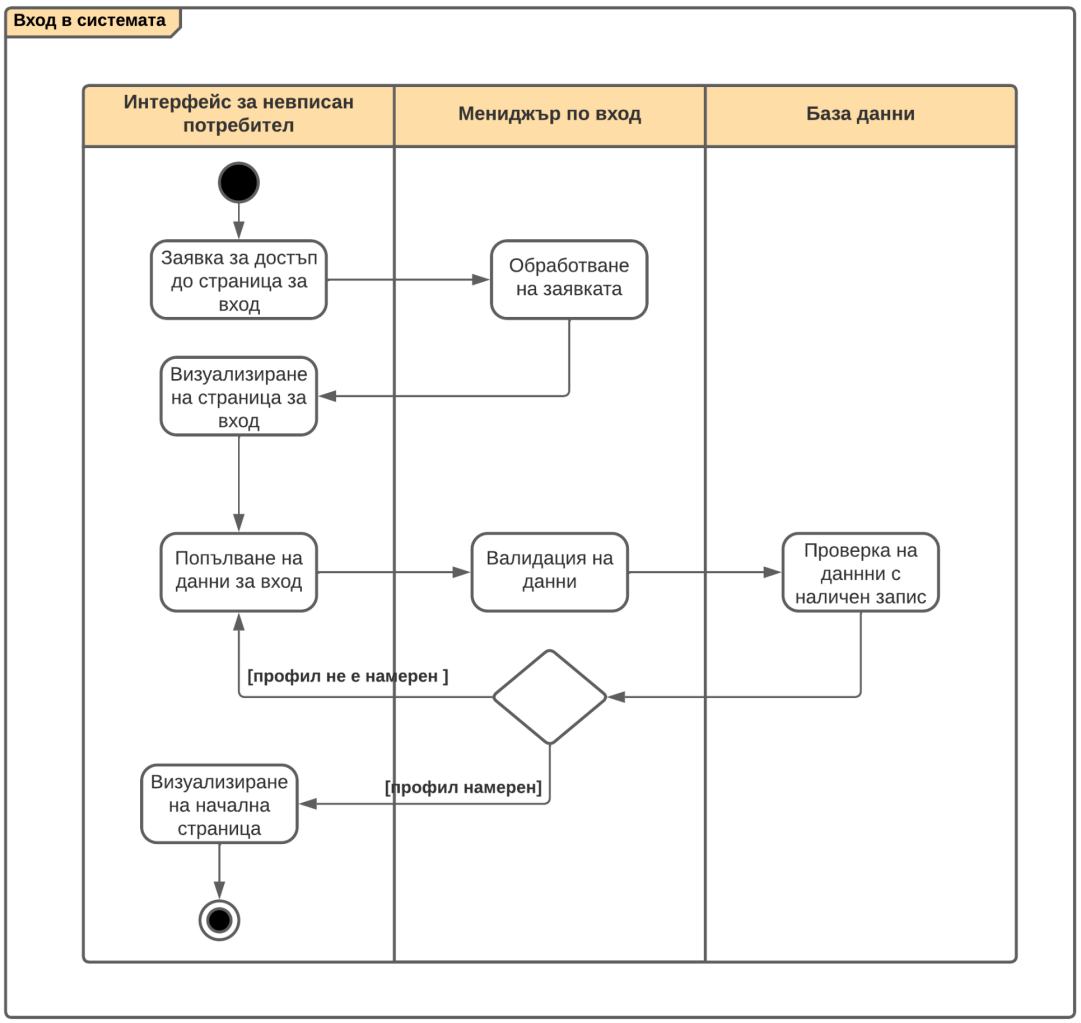

SIMS
Верифициране на протоколи
<<extend>>
Отхвърляне на протокол
<<Include>>
Връщане на преподавател за корекция
Преподавател
Създаване на справки
За студенти
<<Abstract>> Създаване на профил(и)
За преподаватели
Определяне на правила за отделните етапи
<<Extend>>
Администратор на данни
Създаване на кампания
<<Extend>>
За избираеми дисциплини
<<Include>>
Разделяне на етапи (задаване на крайни срокове)
<<Include>>
<<Extend>>
За плащане на семестриални такси
Въвеждане на списък с избираеми дисциплини към кампанията
Редактиране на кампания
<<Extend>>
За избираеми дисциплини
<<Extend>>
Редактиране на етапи
<<Extend>>
<<Extend>>
За плащане на семестриални такси
<<Extend>>
Редактиране на правила за отделните етапи
Добавяне на участници
Премахване на участници
Редактиране на метаданни за курс
<<Extend>>
<<Extend>>
<<Extend>>
Управление на информацията за дисциплина
<<Extend>>
Промяна на критерий за записване на избираема дисциплина
<<Extend>>
Промяна на лимит на избираема дисциплина
Създаване на профил на администратор на данни
<<Include>>
Администратор на системата
Изтриване на профил на администратор на данни

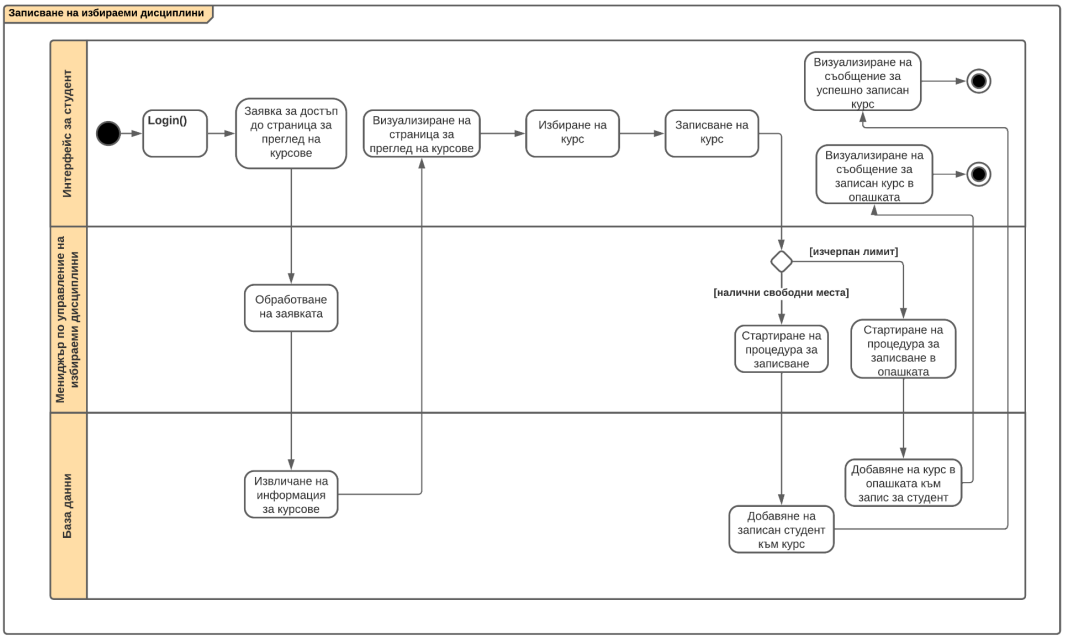
6.2 Диаграми на последователността



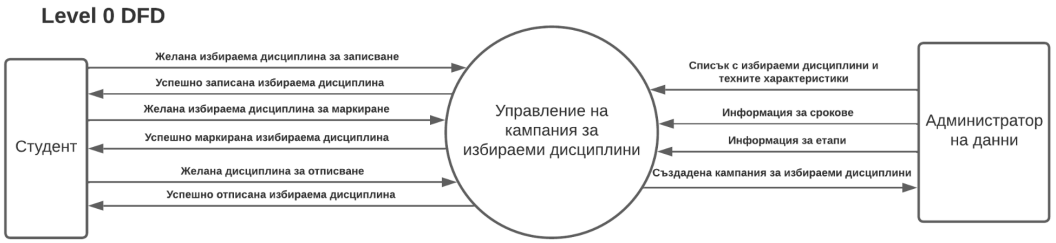


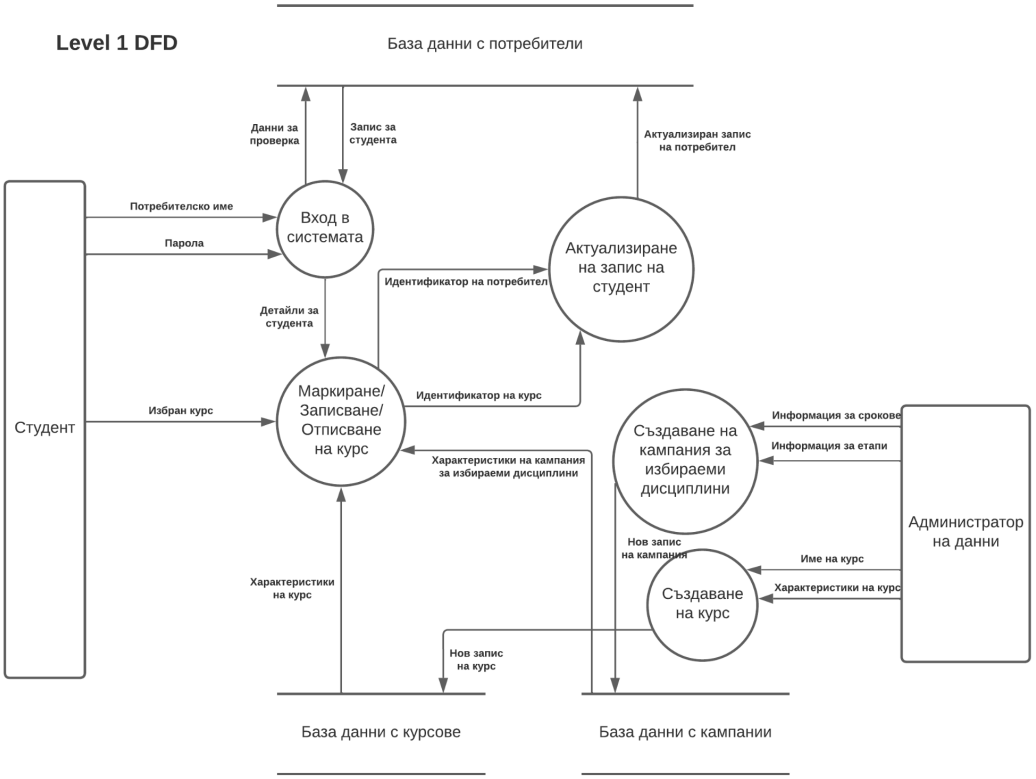
6.3 Диаграми на дейностите



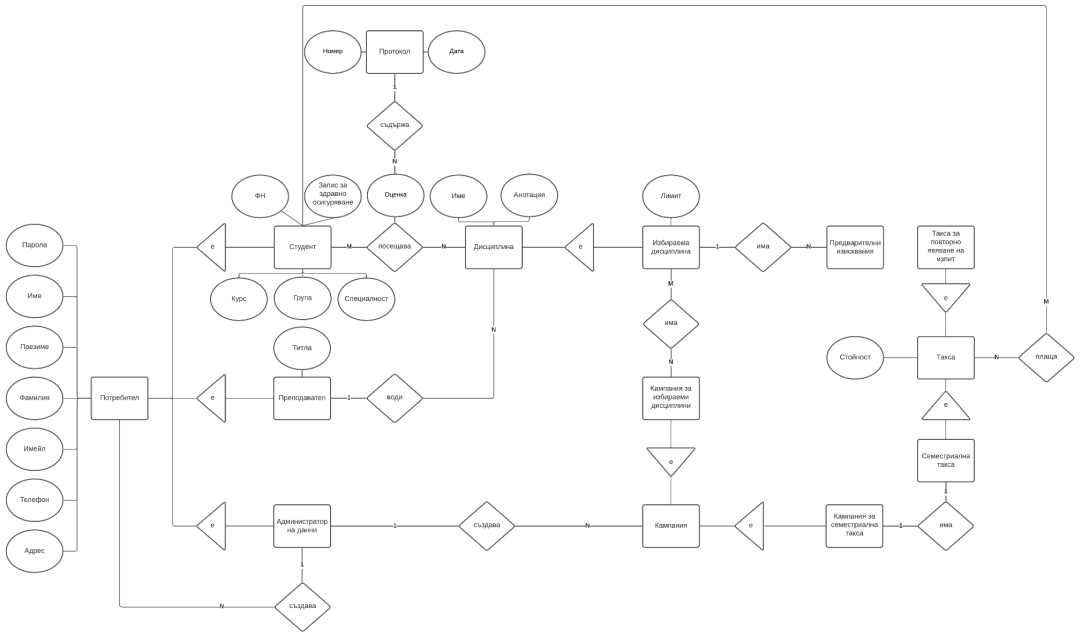


6.4 Диаграма на потока от данни





6.5 Диаграма “Същност - Отношения”



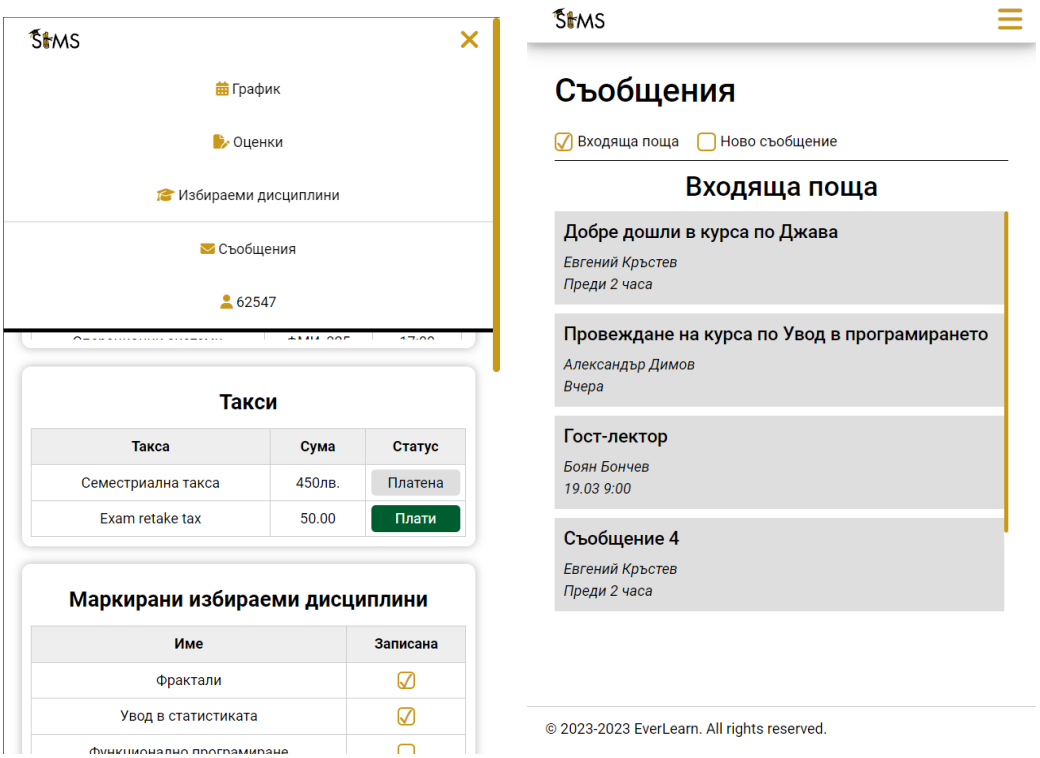
7. Проектиране на интерфейса

Процесът по проектиране на интерфейса беше съпроводен от поредица от интервюта със студенти от различни университети, преподаватели и администратори на подобни системи. Бяха оценени положителните и отрицателните черти, които оказаха влияние върху това какво да се запази от настоящата система СУСИ на Софийския университет и какво да се отстрани или улесни.

Забелязва се, че огромна част от интервюираните лица предпочитат наличието на мобилна версия на една такава система. Именно затова се залага на адаптивния уеб дизайн (responsive web design), с цел подходящото визуализиране на уеб страниците на мобилно устройство.

Една система за управление на студентската информация предполага наличието на поне три различни изгледа - за студенти, преподаватели и администратори. Такова решение беше взето и за нашата система. Различните изгледи ще отговарят на ролята на вписания потребител в системата.

Функционалностите в основното меню на всеки изглед са избрани, така че да отговарят на основните дейности, които се очакват от една такава система. Дизайнът на страниците от гледна точка на различните цветове на елементи, шрифт, могат да бъдат променяни от отделния потребител на системата, с цел постигане на персонализация, която е основен подход, към който се върви и се залага в наши дни от новите, модерни технологии.



8. Използвани технологии

За създаване на прототипа се стъпва основно на HTML, CSS и JavaScript, като технологии, които съвместими с всеки браузър в наши дни.

Проектът е изграден под формата на PHP файлове, като в различни директории се намират изгледите за различните типове потребители на системата. Причината да се използва PHP, е възможността да се включват динамично идентични части като навигация и общи стилове, без да се редактират десетки файлове поотделно, а само един, когато се наложи. Може да разгледате сорс кода на проекта на този адрес: https://github.com/DanielHalachev/SUSI.

След проектирането и създаването на всички страници, файловете бяха интерпретирани от PHP интерпретатор и сведени до чист HTML, CSS и JavaScript, за да могат да бъдат отворени без сървър по време на представянето на проекта. Няма разлика за крайния потребител между HTML и PHP файловете.

Във фаза внедряване, страниците биха останали на PHP и изпълнени на сървър на университета.

За добавяне на икони в менютата и част от уеб страниците е използвана библиотеката Font Awesome, която изисква достъп до интернет в момента на зареждане на страницата. Това не е недостатък, защото така или иначе за достъп до страниците в реална ситуация би се изисквал интернет.

9. Ръководство на потребителя

За преглед на ръководството на потребителя, отворете файла **HCI_UserManual_SIMS_62537_62547_62555_62596.pdf**.