Курсов проект

Дисциплина: Проектиране и интегриране на софтуерни системи

Фаза 3: Реализация на системата

Версия 1.0

Име на студент
Ангел Вардин
Благовеста Костова
Цветелина Петкова
Димитър Манолов
Мартин Василев
Георги Синеклиев

Съдържание

1. Въведение

1.1 Цел

Спецификацията е изготвена с цел да представи реализацията на софтуерната система **„Информационна система за дипломанти и докторанти“**, във връзка с разработването на курсов проект по дисцпилината „Проектиране и интегриране на софтуерни системи“.

1.2 Резюме

За да реализираме софтуерната система „**Информационна система за дипломанти и докторанти**“ сме използвали ASP.NET MVC платформа, която служи за изработване на уеб приложения, използвайки модела Model-View-Controller (MVC). Платформата използва C#, HTML, CSS, JavaScript и бази данни.

1.3 Дефиниции и Акроними

- MVC - ASP.NET MVC е съвременно средство за изграждане на уеб приложения. Платформата може да бъде много лесно тествана и допълвана, защото е изградена от отделни модули, които са изцяло независими едни от други.
- Модел - Моделът представлява множество от класове, които моделират данните, с които работим. Тази концепция е известна още като бизнес логика. В модела може да има и правила, които валидират данните.
- Изглед - Изгледът представлява шаблон, по който се генерира потребителския интерфейс. По–конкретно в десктоп приложенията изгледът е множество от контроли (таблици, падащи менюта), които изграждат видимата част за потребителите.
- Контролери - Контролерите са най-важната концепция от MVC. Те са ядрото на всеки Model-View-Controller продукт и определят логиката, по която приложението работи. Контролерите представляват множество от класове, които приемат заявки от потребителя, обработват данните от заявките и връщат краен резултат. Всеки контролер има едно или повече действия.
- jQuerry – JavaScript библиотека опростява достъпа до всеки елемент на дадена уеб-страница
- jQuerry UI – JavaScript библиотека за визуализация на UI елементи.
- Ugnite IU Free – контроли, разширение на jQuerry UI, който се използват за визуализация на таблици (grid) , datapicker, combobox и др.

2 Използвани Технологии

Технология	Версия
Microsoft Visual Studio	2012
HTML	5
CSS	3
JQuerry	1.10.3
ASP.NET MVC	4
JQuerry UI	1.10
Ugnite IU Free	13.1

- **Razor** - комбинация между html и C#. Позволява реализирането на програмна по-сложна спрямо обикновения html, спестяване на излишно писане на html. Това се постига, защото страницата е динамична т.е. се рендира по време на изпълнението (runtime).

- **Ugnite UI Free** - контроли, разширение на jQuerry UI, който се използват за визуализация на таблици (grid) , datapicker, combobox и др. По този начин се спестява писане на допълнителен css и javascript за тяхната визуализация както и много допълнителни функции. Освен това те съдържат server wrappers като по този начин могат да се използват чрез razor синтаксиса, който поддържа mvc шаблона в Visual Studio 2012.

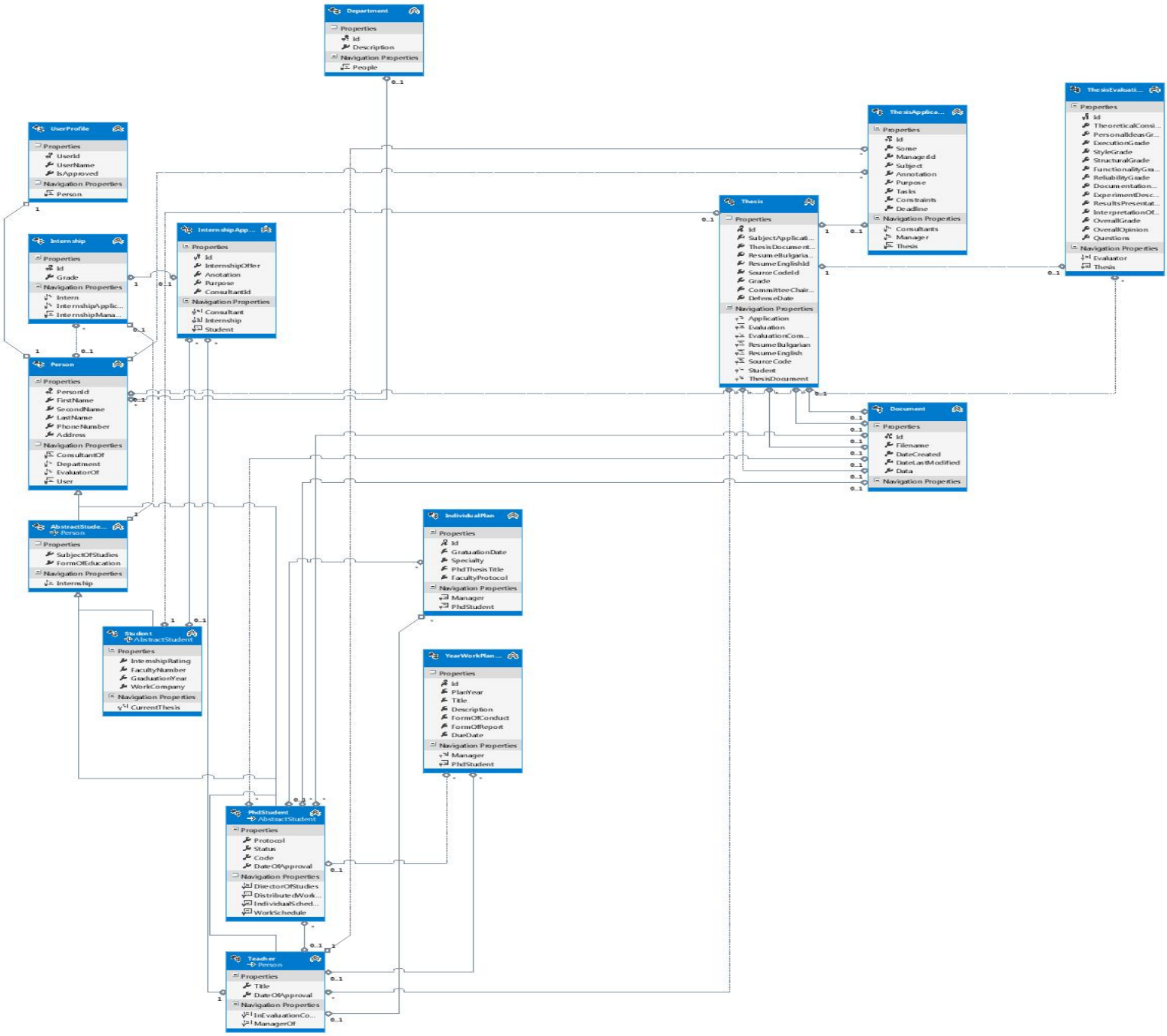
- **Css** - Cascading Style Sheets е език за описание на стилове - използва се основно за описване на представянето на документ, написан на език за маркиране. Най-често се използва заедно с HTML. CSS позволява да се определя как да изглеждат елементите на една HTML страница - шрифтове, размери, цветове, фонове, и др.

- **Unit of Work** - а рхитектурен шаблон, който позволява да се работи не направо с контекста на данните, а с рехните репозиторита. По този начин може да се смени контекста на данните, без да има промени бо бизнес логиката, което улеснява тестването и подмянятя на БД.

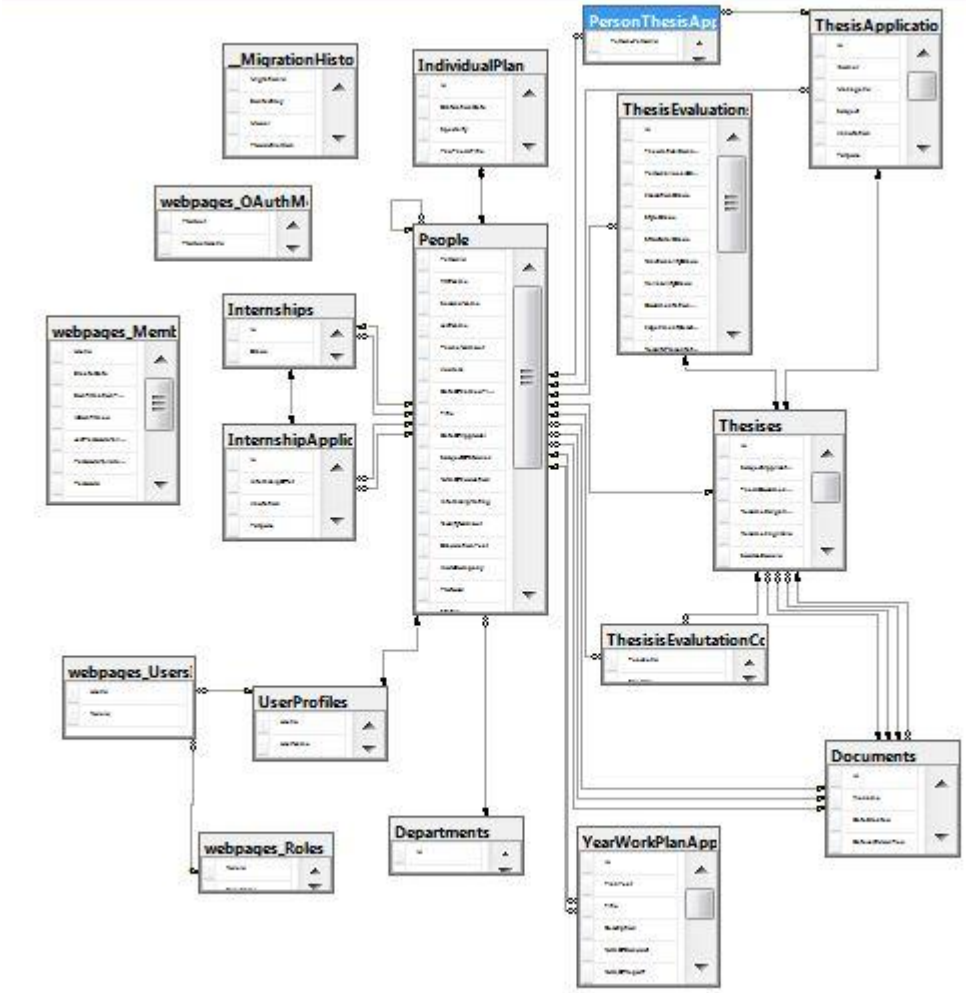
3 Реализация на базата от данни

Използвали сме MSSQL Server за съхранение на базата данни. Технологията Entity Framework позволява техниката Code First, която създава модел и таблици на базата данни по съществуващи класове на приложението, наречени Entity-та. Това позволява гъвкавосто по време на проектиране на приложението(Entity класовете могат да се променят по време на разработване като има механизъм за мигриране на базата).

Entity класовете ни представляват логическото разделение на типовете данни.



Следва диаграма на схемата на базата данни:



Както виждаме, Entity Framework асоцииира нетрадиционно класовете от моделите ни към таблици. Например Entity-тата **Person, Student, AbstractStudent, Teacher, phdStudent** са представени в една таблица. Така няма да има нужда от join операции в базата когато е необходимо взаимодействие между студенти, и учители(което е често срещана операция). Недостатък на текущото представяне е заема повече памет(някои колони, които са задължителни само за Student, ще бъдат null за други entity-та). Тази допълнителна памет обаче не е фатална, тъй като системата не би трябвало да поддържа голям брой потребители.

4 Реализация на Бизнес Логиката

За реализиране на бизнес логиката сме разделили „**Информационна система за дипломанти и докторанти**“ на следните модули:

4.1 Модул Потребители

Предоставя функционалности за управление на акаунтите и личната информация на потребителите на системата

Основната функционалност на Модула е имплементирана в контролерите: Account, UserInfo , в администраторският панел User и в преподавателския панел TeacherPersonalInformation.

AccountController

В Account контролера се съдържа функционалността за създаване но нов потребителски профил, влизане и излизане от сайта и промяна на парола. Основни методи в него са:

public ActionResult Register(RegisterModel model)

Извиква view-то с регистрационната форма

public ActionResult RegisterStudent(RegisterModel model, StudentInfo student)

Регистриране на нов акаунт на студент като се взимат данни от съответната регистрационна форма съответната

public ActionResult RegisterTeacher(RegisterModel model, TeacherInfo teacher)

Регистриране на нов акаунт на учител като се взимат данни от съответната регистрационна форма съответната

public ActionResult RegisterPhdStudent(RegisterModel model, PhdStudentInfo student)

Регистриране на на нов акаунт на докторант като се взимат данни от съответната регистрационна форма съответната

public ActionResult Manage(LocalPasswordModel model)

Промяна на парола на потребителител на системата.

public ActionResult Login(LoginModel model, string returnUrl)

Влизане в личния профил на потребител.

public ActionResult Logoff()

Излизане от профила.

public ActionResult RegisterStudent(RegisterModel model, StudentInfo student)

Метод, който зарежда регистрационната форма. В зависимост от избраната форма от падащ списък се зареждат специфичен изглед, който е характерен за съответната роля

UserInfoController

В този контролер е имплементирана функционалност за промяна на персонална информация за потребител.

public ActionResult Edit()

Този метод според ролята на потребителя извиква изглед, който е подходящ за неговата роля.

public ActionResult EditTeacherInfo(EditTeacherInfo model),

public ActionResult EditStudentInfo(EditStudentInfo model),

public ActionResult EditPhdStudentInfo(EditPhdStudentInfo model)

Методи, които съхраняват променената информацията за потребителя в зависимост от неговата роля

TeacherPersonalInformation

В този контролер е имплементирана функционалност за промяна на персонална информация на преподавател в Teachers панела .

public ActionResult Edit()

Този метод извиква форма за редактиране на персоналната информация на преподавател.

public ActionResult EditTeacherInfo()

Метод, които съхраняват променената информацията за потребителя

Функции на администратора

Администраторът на системата може да одобрява новите потребители, да добавя и изтрива акаунти и да добавя нови катедри. Контролерите в този панел са:

UserController

В този контролер има фунционалностти за за одобряване и изтриване на потребители.

public ActionResult Index()

Този метод извиква форма, която съдържа таблица с всички потребители в системата, която взима информация от метода **public ActionResult AllUserList ()**

public ActionResult ApproveUser(int user)

Одобряване на нов потребител. Този метод се извиква чрез Ajax заявка и като резултат връща съобщение до заявката за успеха на операцията**.**

public ActionResult AllUsers()

Одобряване на заявки на ново регистрирани потребители. В изгледа , който се зарежда при извикването на метода съдържа таблица, която взима информация от метода **public ActionResult UserList().**

public ActionResult DeleteUser(int user)

Изтрива потребителя от системата. Този метод се извиква чрез Ajax заявка и като резултат връща съобщение до заявката за успеха на операцията**.**

public ActionResult Details(int userId)

Връща страница с информация за потребителя.

public ActionResult SendMail(int user)

Изпраща имейл за одобряване на нов потребител.

DepartmentController

Чрез този контролер се администрират катедрите в системата.

public ActionResult Index ()

Зарежда изглед с всички катедри в системата в таблица, която взима информацията си от метода **public ActionResult AllDepartments()**

public ActionResult UpdatingSaveChanges() T

Този метод се извиква ajax при съхраняване на промените в информацията за катедрите. Той взима транзакциите, които се създават при всяка промана на в таблицата и в зависимост от типа на транзакцията извършва съответните промени.

4.2 Модул Дипломна Работа

Предоставя функционалности свързвани със създаването и обработването на дипломна работа.

Основната функционалност на Модула е имплементирана в контролерите: Thesis и ThesisInfo. Процесът на дипломиране на един студент протича в следните стъпки:

1. Студентът попълва форма, която служи като заявление за дипломна работа.
2. Дипломния ръководител разглежда заявлението и го одобрява
3. Студентът може да започне работа - има право да качва файлове(анотация, дипломна работа, сорс код и др)
4. Рецензентът на дипломната работа може да добави рецензия.
5. Дипломният ръководител назначава дата за защита и изпитна комисия.

ThesisController обхваща функциите на дипломанта , а ThesisisInfoController - тези на дипломният ръководител.

ThesisController

public ActionResult Index()

Зарежда главната страница. Там е показано текущото състояние на дипломната работа и предоставя възможности за качване и сваляне на файлове, както и експорт на някои документи.

public ActionResult Create(ThesisApplication application, FormCollection collection)

Добява в базата нова дипломна работа по подадени от уеб-форма данни. Там се посочват дипломен ръководител, консултанти и друга начална информация)

private JsonResult UploadFileForPerson(int personId, UploadFileType fileType)

Ъплоудва файл на студент към сървъра. Там се съхраняват анотации, дипломни работи и сорс-код.

public FileResult DownloadDocument(int documentId)

Сваля файл от сървъра към клиента.

public FileResult GetThesisApplicationDocument()

ThesisisController

public ActionResult Index()

Показва основната страница. Тук има възможност за различни видове справки на дипломираните студенти(от дата до дата), както и филтрирането им по дипломен ръководител и реценцент.

public ActionResult ThesisisAssignedToMe()

public ActionResult NewUserList()

Помощни методи за изчисляване и филтриране на информацията за дипломирани студенти.

public ActionResult AddThesisEvaluation(int studentId)

Добавяне на рецензия към дипломна работа. Право на рецензия имат консултантите, които студентът е избрал.

public ActionResult ApproveThesisis()

Показва страница с информация за дипломните работи, които чакат одобрение. От там могат да се разглеждат дипломни работи, както и да се променя техния статус.

public ActionResult ThesisEvaluationDetails(int studentId)

Показва Детайли за съответната дипломна работа.

public FileResult getThesisEvaluationDocument(int studentId)

Служи за експорт на рецензия на дипломна работа.

4.3 Модул „Преддипломент Стаж“

Предоставя функционалности за провеждане и оценяване на преддиплоемен стаж.

Основната функционалност на модула е имплементирана в контролера InternshipController и InternshipInfoController. Процесът на въвеждане и оценяване на преддипломен стаж протича по следния начин:

1. Студентът попълва форма, която служи като заявление за преддипломен стаж
2. Преподавател разглежда формата и я одобрява
3. Студентът въвежда оценката, която е получил на стажа.
4. Преподавател разглежда оценката и ако прецени, че всичко е наред я одобрява

InternshipController

Той обработва информацията при студента.

public ActionResult Index() - показва основната страница за студента. Тук той има възможност ако няма стаж да подаде заявка за такъв, а ако има да го разгледа или ако съответно е приет за този стаж и е получил оценка да я въведе.

public ActionResult AddInternship(InternshipApplication internshipApplication, FormCollection collection) - показва форма, през която студентът въвежда информацията за стажа, който иска да заяви.

InternshipInfoController

Той обработва информацията при преподавателя.

public ActionResult Index() - показва менюто със стажове чакащи за удобрение на екрана

public ActionResult AcceptInternship(Internship internship, FormCollection collection) - чрез метода преподвателят може да одобри стаж или оценка за стаж на даден студент.

4.4 Модул „Докторанти“

Предоставя функционалности свързвани с индивидуален план, работен план по години и общ работен план на потребители-докторанти. Също така този модул отговаря за справките за докторанти.
Общият работен план представлява съвкупността от работните планове по години. Поради този факт функционалността може да имплементира като се създаде форма за работен план по години и след това да се вземат работните планове за всички години и да се състави общият.
Основната функционалност на Модула е имплементирана в контролерите: PhdStudent, WorkPlan и IndividualPlan.
Процесите в модула са едностъпкови – това, което се въведе от докторанта се записава без разглеждаде или одобрение от ръководител.

PhdStudentController

public ActionResult Index()
Възможност за навигация към различните функионалности.

Справки. Тъй като тази функционалност е изцяло свързана с докторанти, но самите те като потребители нямат достъп до нея, се ограничава достъпа с потребителски роли. Използват се четири метода за извличане на справки от базата.

public JsonResult GetAllGraduatePhdsInRange(string fromDate, string toDate)
Прави справка за докторантите, който са завършили от дата до дата.
public JsonResult GetAllGraduatePhdsInRangeWithManager (string fromDate, string toDate, int teacherId)
Прави справка за докторантите, който са завършили от дата до дата и имат определен научен ръководител.
public JsonResult GetAllNotGraduatePhdsInRange(string fromDate, string toDate)
Прави справка за текущите докторантите от дата до дата.
public JsonResult GetAllNotGraduatePhdsInRangeWithManager (string fromDate, string toDate, int teacherId)
Прави справка за текущите докторантите от дата до дата, които имат определен научен ръководител.

WorkPlanController

public ActionResult Index()
Зарежда въдедените до момента работни планове. Представянето е в табличен вид. Предоставя достъп до другите функционалности на контролера – създавене на нов, редактиране или изтриване на текущ, експорт към текстов файл.

public ActionResult Create(FormCollection collection)
Добява в базата нов работен план за година по подадени от уеб-форма данни.

public ActionResult Edit(int id, FormCollection collection)
Опреснява запис от базата с нови стойности.

public ActionResult Delete(int id)
Изтрива запис на работен план от базата.

public FileResult ExportData()
Експортира данните от дадения работен план към тексов файл.

IndividualPlanController

public ActionResult Index()
Основна функционалност – показва въведния до момента работен план. Предоставя възможност за изтриване, редактиране и експорт.

public ActionResult Create(FormCollection collection)

Създава нов индивидуален план по въвдените във формата полета.

public ActionResult Edit(int id, FormCollection collection)
Промяна на вече съществуващ индивудуален план.

public ActionResult Delete(int id)
Изтриване на индивудуалния план.

public FileResult ExportData()
Експортира данните от индивидуалния план към текстов файл.

5 Реализация на поребителския интерфейс

Потребителския интерфейс е View-частта от MVC модела. За визуализация са използвани web-технологиите описани в първата част на документа.

Account

1. Login.cshtml - Това View предоставя достъп на потребителя до панела за вход в системата с опция за регистрация, ако потребителя няма акаунт.

2. Manage.cshtml - View, което дава възможност на потребителя да редактира своя акаунт. Редакцията се състои единствено в промяна на текущата парола с нова.

3. Register.cshtml - View, предоставящо възмовност за регистрация на нови потребители в зависимост от типа потребител (студент, докторант, учител) - регистрационна форма.

UserInfo

1. EditStudent.cshtml - Това View предоставя възможност на студента да редактира своята лична информация.

2. EditPhdStudent.cshtml - Това View предоставя възможност на докторанта да редактира своята лична информация.

3. EditTeacher - Това View предоставя възможност на учителя да редактира своята лична информация.

Individual Plan

1. Create.cshtml - Това View е предназначено за докторанти и предоставя възможност за създаване на индивидуален работен план.

2. Delete.cshtml - View, предназначено за изтриване на вече създаден работен план.

3. Edit.cshtml - View, предназначено за редактиране на вече създаден работен план.

4. Index.cshtml - View, което показва въведния до момента работен план. Предоставя възможност за изтриване, редактиране и експорт.

Internship

1. AddInternship.cshtml - Това View е предназначено за студенти и дава възможност за добавяне на преддипломен стаж.

2. Index.cshtml -Страница, която визуализира дипломния стаж , ако има такъв и препдаща евентуално към View-то за добавяне на дипломен стаж .

PhdStudent

1. Index.cshtml - View, което дава възможност на докторант да достъпи работния план по години или индивидуалния план.

Thesis

1. Index.cshtml - Това е View, което препраща студентите към предлагане на тема за дипломна работа ако даден студент няма одобрена такава; ако даден студент има вече удобрена тема в този прозорец той може да добавя различни документи или архиви свързани с дипломната работа, предоставена е и възможност за експорт на документа.

2. Create.cshtml - View, което предоставя възможност на студента да предлойи тема за дипломна работа, да избере преподавател и комисия

Admin/Department

1. Index.cshtml - Страница, предоставяща възможност за преглед на текущите катедри и тяхното редакриране.

Teachers/ThesisisInfo

1. AddEvaluationCommission.cshtml - View, което дава възможност на преподавателя да свърже студента и неговата дипломна работа с комислия за оценяването и.

2. AddThesisEvaluation.cshtml - View, което предоставя прозорец за въвеждане на оценка към дипломна работа на студент.Предоставя възможност за детайлно описание на рецензия към дипломна работа.

3. ApproveThesis.cshtml - Страница с информация за дипломните работи, които чакат одобрение. Предоставя се възмойност за разглеждане на дипломни работи, както и промяна на техния статус.

4. Index.cshtml - Предоставя преглед на дипломните работи и студентите, асоциирали се с даден преподавател
5. ThesisEvaluationDetails.csthml -. Показва Детайли за съответната дипломна работа.

Teachers/InternshipInfo

1. Index.cshtml - View, предназначено за учител. Използва се за визуализране на заяките от страна на студенти за предизборен стаж

2. AcceptInternship.cshtml - View, да удобрение на предизборния стаж

Teachers/TeacherPersonalInformation

1. EditTeacher - Това View предоставя възможност на учителя да редактира своята лична информация.

6 Внедряване на системата

За да внедрите на системата трябва да я деплойните на сървър или в облак, който може да стартира програми компилирани на .net 4.0, да разполага с MSSQL Server и да има инсталиран уеб сървъра IIS 7.

За да я деплойните трябва да смените в web cofig файла пътя до БД. После трябва да мигрирате Базата данни от конкеста (DbContextIpl.cs) като в нея има регистрирани няколко потребителя. И най-накрая трябва да се компилира системата.