**Аннотация**

В рамках курсового проекта разработана программная система для дистанционного обучения студента. Система представляет собой веб приложение, позволяющее хранить и обрабатывать информацию, касающуюся обучения.

Пояснительная записка содержит 24 листов. Включает в себя: 9 рисунков, 1 приложение.

As part of the course project, a software system for distance learning of the student has been developed. The system is a web application that allows you to store and process information related to training.

Explanatory note contains 24 sheets. Includes: 9 figures, 1 appendice.

*Федеральное государственное бюджетное образовательное учреждение высшего образования*

*«Владимирский государственный университет*

*имени Александра Григорьевича и Николая Григорьевича Столетовых»*

*Кафедра информационных систем и программной инженерии*

***ПОЯСНИТЕЛЬНАЯ ЗАПИСКА***

***к курсовому проекту по дисциплине   
"Технологии программирования"***

***на тему***

*Проектирование и разработка программной системы   
«Дистанционное обучение студента»*

*Выполнил: студент гр. ИСТ-116*

*Никифоров М.Д.*

*Принял: доц. кафедры ИСПИ*

*Вершинин В.В.*

*Владимир, 2018*

**Содержание**

[1. ВВЕДЕНИЕ 3](#_Toc535818759)

[2. Постановка задачи 4](#_Toc535818760)

[3. Описание предметной области 5](#_Toc535818761)

[3.1. Общие сведения о предмете автоматизации 5](#_Toc535818762)

[3.2. Словарь предметной области 5](#_Toc535818763)

[3.3. Сценарий взаимодействия пользователя с системой 6](#_Toc535818764)

[4. проектирование системы 7](#_Toc535818765)

[4.1 Диаграмма прецедентов 7](#_Toc535818766)

[4.2 Диаграмма классов 9](#_Toc535818767)

[4.3 Диаграмма состояний 10](#_Toc535818768)

[4.4Диаграмма базы данных 11](#_Toc535818769)

[5.1 Принципы организации проекта 11](#_Toc535818775)

[5.2 Реализация поддержки различных типов пользователей 11](#_Toc535818776)

[5.3 Реализация поддержки различных справочников 12](#_Toc535818777)

[5.4 Пользовательский интерфейс 12](#_Toc535818778)

[6. заключение 14](#_Toc535818779)

[7. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc535818780)

[ПРИЛОЖЕНИЕ А Исходный код программной системы 16](#_Toc535818781)

**1. ВВЕДЕНИЕ**

В данном курсовом проекте производится проектирование и реализация программной системы для дистанционного обучения студента.

В ходе выполнения курсового проекта, требуется решить следующие задачи:

* анализ предметной области;
* разработка проекта программной системы;
* разработка прототипа программной системы;
* реализация программной системы.

Результатом выполнения курсового проекта будет работающее программное обеспечение.

В ходе выполнения курсового проекта выполнена разработка проекта информационной системы с использованием языка UML, реализация диаграмм с использованием среды Microsoft Office Visio, реализация программного обеспечения с использованием технологий разработки ASP .NET MVC Entity Framework. Среда программирования – Microsoft Visual Studio 2015.

1. **Постановка задачи**

Необходимо последовательно выполнить этапы анализа, проектирования и реализации системы;

Анализ предметной области. Результатом данного этапа работы является описание бизнес-процесса и обработки данных.

Проектирование системы. Для проектирования системы необходимо использовать язык UML и включать в себя следующие UML-диаграммы:

* диаграмма прецедентов;
* диаграмма классов;
* диаграмма состояний;

Также этап проектирования включает схему базы данных в виде ER-диаграммы логической моделей данных.

Программная реализация.

Минимальный набор функций:

1. работа со справочниками автобусов и маршрутов;

2. формирование путевого листа;

3. учет данных по автобусам.

Система должна быть реализована на языке C# в виде веб-приложения с использованием технологий ASP .NET MVC.

1. **Описание предметной области**
   1. **Общие сведения о предмете автоматизации**

Темой данного курсового проекта является программная система учёта дистанционного обучения в образовательном учреждении. Подобная программная система позволяет образовательным учреждениям расширить функциональные возможности как преподавателей, так и учеников. Пользователями системы дистанционного обучения станут студенты и преподаватели, а также руководство учебного заведения.

Пользуясь программной системой дистанционного обучения, персонал так же получит широкие возможности для планирования и контроля учебного процесса. В системе будут реализованы следующие функции: возможность написать или прикрепить ответ к заданию для студента, управление пользователями и ролями для администратора, возможность управления курсами для преподавателей.

Администрация образовательного учреждения с помощью программной системы сможет добавлять преподавателей и студентов в базу данных, создавать списки новых и текущих групп и формировать для них удобное расписание. Система предусматривает три роли для администрации образовательного учреждения: администратор, студент и преподаватель.

Регистрация новых пользователей производится персоналом учебного заведения.

* 1. **Словарь предметной области**

Пользователь – какой-либо зарегистрированный участник программной системы. Каждый пользователь имеет единственно определенную роль.

Студент – роль, при которой пользователь, который имеет функцию просмотра своего каталога курсов, а также принадлежит к какой-либо группе.

Преподаватель – роль, при которой пользователь имеет доступ к каталогу курсов. Регистрацию в системе производит администратор.

Администратор – роль, при которой пользователь может подтверждать внесённые изменения и управлять создание пользователей и предметов.

* 1. **Сценарий взаимодействия пользователя с системой**

В приложении используется разделение по ролям.

Пользователь с ролью *Студент* может:

* Просматривать лекции, темы в них и тесты
* Отвечать на тесты

Пользователь с ролью *Преподаватель* может:

* Добавлять лекции и тесты по ним
* Редактировать лекции и тесты
* Удалять лекции и тесты

Пользователь с ролью *Администратор* может также:

* Удаление пользователей

Гость имеет возможность:

* Авторизоваться;
* Зарегистрироваться;

1. **проектирование системы**
   1. **Диаграмма прецедентов**

Объект информатизации исследуется в границах одного образовательного учреждения. Действующими лицами (в нотации UML – актерами) являются: Студент, Преподаватель, Администратор и Гость. Ниже приведен перечень и описание некоторых прецедентов в проектируемой системе.

***Название:*** «Просмотр тестов»

***Предусловие:*** пользователь имеет доступ к веб-сайту программной системы.

***Действующее лицо:*** студент

***Основной поток:*** Действующее лицо открывает окно приложения, отображающую информацию, заполненную в каталоге. Пользователь может просматривать доступные ему тесты.

***Название:*** «Удаление пользователей»

***Предусловие:*** персонал имеет доступ к веб-сайту программной системы.

***Действующее лица:*** администратор.

***Основной поток:*** действующее лицо открывает форму управления каталогом пользователей и выполняет необходимые ему действия.

***Название:*** «Редактирование лекций»

***Предусловие:*** пользователь имеет доступ к веб-сайту программной системы.

***Действующее лицо:*** Преподаватель

***Основной поток:*** Действующее лицо открывает форму управления лекциями. Переходит на страницу редактирования и выполняет необходимые ему действия.

***Альтернативный поток:*** Действующее лицо по какой либо причине решило не завершать редактирование лекции. Лекция осталась без изменений.

Диаграмма прецедентов представлена на рисунке 1.

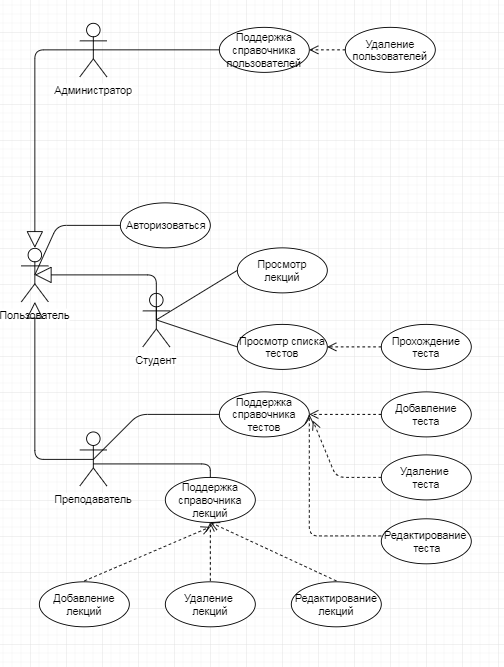


Рисунок 1. Диаграмма прецедентов

* 1. **Диаграмма классов**

Для проектирования структур и моделей данных программной системы была разработана диаграмма классов, представляющая основные сущности системы. Диаграмма классов представлена на рисунке 2.

Программной система реализована частично по паттерну проектирования MVC, методы доступа и взаимодействия с базой данных будут реализованы в классах DAO. На данном уровне абстракции эти методы исключены из диаграммы.

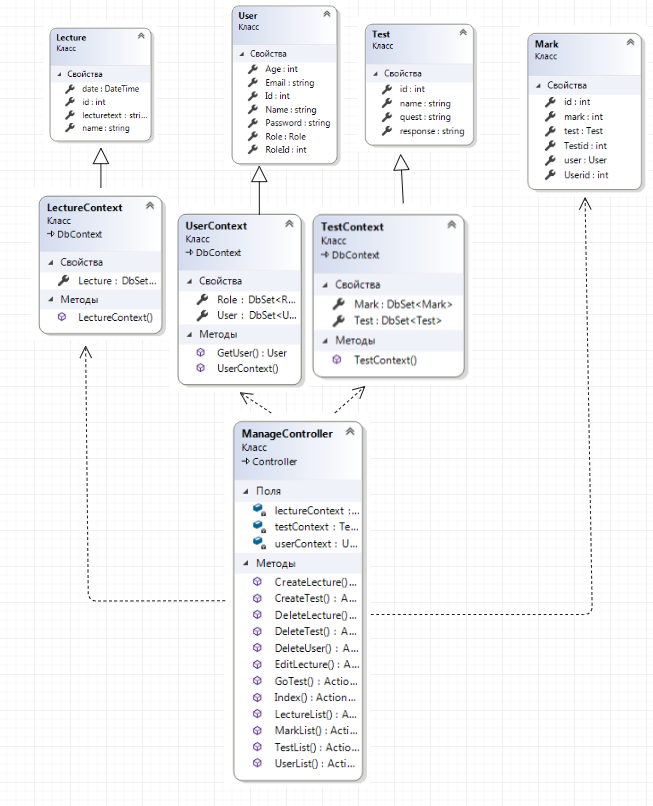


Рисунок 2. Диаграмма классов

* 1. **Диаграмма состояний**

Состояние объекта «Тест».

Тест создаётся лектором, позже проходится студентом и проверяется системой, после добавления результатов в таблицу он готов к удалению.

Диаграмма состояний представлена на рис.3.



Рисунок 3. Диаграмма состояний объекта «Тест».

**4.4Диаграмма базы данных**

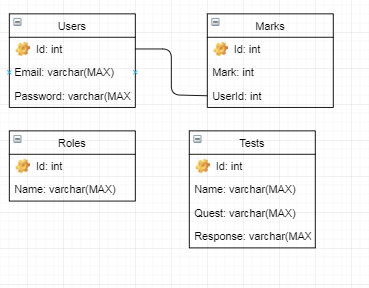


Рисунок 5. Схема базы данных.

**реализация системы**

5. 1. **Принципы организации проекта**

Разрабатываемая система построена частично на архитектуре MVC с использованием технологии Entity Framework. Система представляет собой веб-приложение, и, таким образом, разделяется на три части: контроллер, модель, представление.

* 1. **Реализация поддержки различных типов пользователей**

Для реализации регистрации, аутентификации и авторизации пользователей были использованы стандартные методы Microsoft Identity.

Методы и представления имеют ограниченный доступ. Пользователям, в зависимости от роли, доступны различные функции. Это обеспечивается ограничениями, реализованными с помощью встроенных средств.

* 1. **Реализация поддержки различных справочников**

С помощью различных форм пользователи с ролью преподаватель могут взаимодействовать с такими справочниками как: справочник лекций и справочник тестов.

* 1. **Пользовательский интерфейс**

Пользовательский интерфейс представлен на Рисунках 5-8.

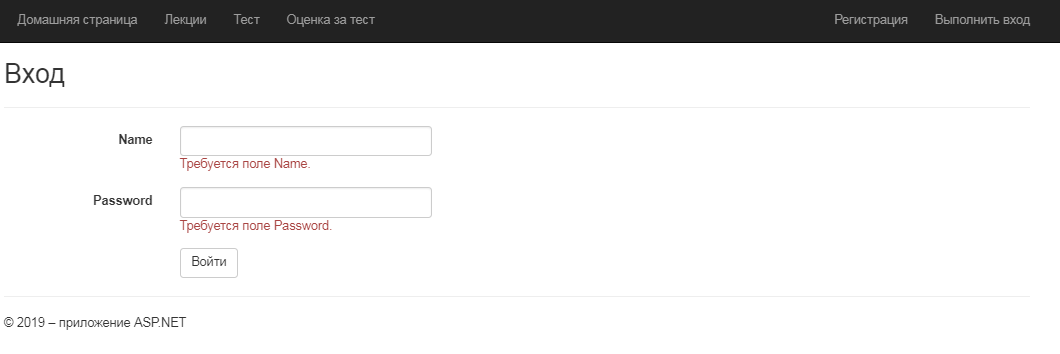


Рисунок 5. Страница входа

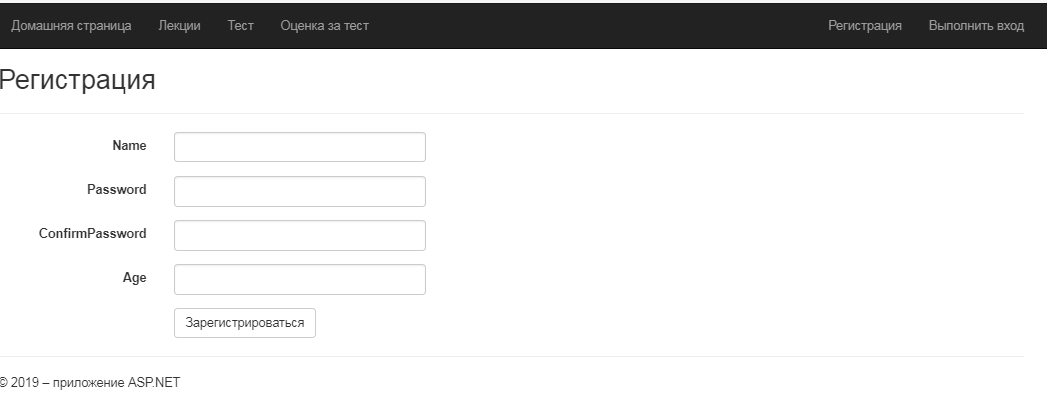


Рисунок 6. Страница регистрации



Рисунок 7. Справочник пользователей

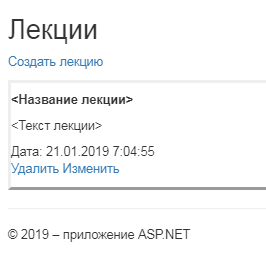


Рисунок 7. Лекции.

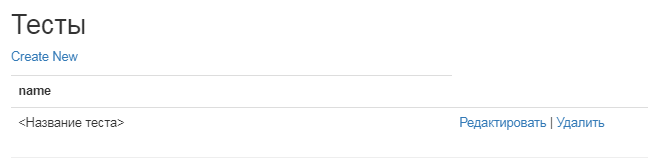


Рисунок 8. Тест.

* 1. **заключение**

В ходе курсового проекта были получены навыки создания и поддержки программных систем. Выполнено проектирование и реализация программной системы для автобусного предприятия.

В ходе выполнения курсового проекта были решены следующие задачи:

* анализ предметной области;
* разработка проекта программной системы;
* разработка прототипа программной системы;
* реализация программной системы.

Результатом выполнения курсового проекта стало работающее программное обеспечение.

В ходе выполнения курсового проекта выполнена разработка проекта информационной системы с использованием языка UML, реализация диаграмм с использованием среды Microsoft Office Visio, реализация программного обеспечения с использованием технологий разработки ASP .NET MVC Entity Framework. Среда программирования – Microsoft Visual Studio 2017

* 1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Липаев В.В. Проектирование программных систем. М.: Высш. шк, 1990.

2. Буч Г. Объективно-ориентированное проектирование / Пер. с анг. Канкорд, 1996.

3. Майерс Г. Надежность программного обеспечения. М.:Мир, 1980

4. Калянов Г.Н. CASE структурный системный анализ (автоматизация и применение.) – М.: «ЛОРИ», 1996.

**ПРИЛОЖЕНИЕ А Исходный код программной системы**

**Модели:**

**Lecture.cs:**

namespace Student.Models

{

public class Lecture

{

public int id { get; set; }

public string name { get; set; }

public string lecturetext { get; set; }

public DateTime date { get; set; }

}

}

**LectureContext.cs**

using System.Data.Entity;

namespace Student.Models

{

public class LectureContext : DbContext

{

public LectureContext() :

base("Database1")

{ }

public DbSet<Lecture> Lecture { get; set; }

}

}

**Mark.cs:**

namespace Student.Models

{

public class Mark

{

public int id { get; set; }

public int mark { get; set; }

public User user { get; set; }

public int Userid { get; set; }

public Test test { get; set; }

public int Testid { get; set; }

}

}

**Models.cs:**

using System.ComponentModel.DataAnnotations;

namespace Student.Models

{

public class LoginModel

{

[Required]

public string Name { get; set; }

[Required]

[DataType(DataType.Password)]

public string Password { get; set; }

}

public class RegisterModel

{

[Required]

public string Name { get; set; }

[Required]

[DataType(DataType.Password)]

public string Password { get; set; }

[Required]

[DataType(DataType.Password)]

[Compare("Password", ErrorMessage = "Пароли не совпадают")]

public string ConfirmPassword { get; set; }

[Required]

public int Age { get; set; }

}

}

**Test.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace Student.Models

{

public class Test

{

public int id { get; set; }

public string name { get; set; }

public string quest { get; set; }

public string response { get; set; }

}

}

**TestContext.cs:**

using System.Data.Entity;

namespace Student.Models

{

public class TestContext : DbContext

{

public TestContext() :

base("Database1")

{ }

public DbSet<Test> Test { get; set; }

public DbSet<Mark> Mark { get; set; }

}

}

**Usercs**:

namespace Student.Models

{

public class User

{

public int Id { get; set; }

public string Email { get; set; }

public string Password { get; set; }

public int Age { get; set; }

public string Name { get; set; }

public int RoleId { get; set; }

public Role Role { get; set; }

}

public class Role

{

public int Id { get; set; }

public string Name { get; set; }

}

}

**UserContext.cs:**

using System;

using System.Data.Entity;

using System.Linq;

namespace Student.Models

{

public class UserContext : DbContext

{

public UserContext() :

base("Database1")

{ }

public DbSet<User> User { get; set; }

public DbSet<Role> Role { get; set; }

public User GetUser(String name)

{

return User.FirstOrDefault(u => u.Email == name);

}

}

}

**AccountController.cs:**

using Student.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using System.Web.Security;

namespace Student.Controllers

{

public class AccountController : Controller

{

public ActionResult Login()

{

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Login(LoginModel model)

{

if (ModelState.IsValid)

{

// поиск пользователя в бд

User user = null;

using (UserContext db = new UserContext())

{

user = db.User.FirstOrDefault(u => u.Email == model.Name && u.Password == model.Password);

}

if (user != null)

{

FormsAuthentication.SetAuthCookie(model.Name, true);

return RedirectToAction("Index", "Home");

}

else

{

ModelState.AddModelError("", "Пользователя с таким логином и паролем нет");

}

}

return View(model);

}

public ActionResult Register()

{

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Register(RegisterModel model)

{

if (ModelState.IsValid)

{

User user = null;

using (UserContext db = new UserContext())

{

user = db.User.FirstOrDefault(u => u.Email == model.Name);

}

if (user == null)

{

// создаем нового пользователя

using (UserContext db = new UserContext())

{

db.User.Add(new User { Email = model.Name, Password = model.Password, Age = model.Age, RoleId = 2 });

db.SaveChanges();

user = db.User.Where(u => u.Email == model.Name && u.Password == model.Password).FirstOrDefault();

} // если пользователь удачно добавлен в бд

if (user != null)

{

FormsAuthentication.SetAuthCookie(model.Name, true);

return RedirectToAction("Index", "Home");

}

}

else

{

ModelState.AddModelError("", "Пользователь с таким логином уже существует");

}

}

return View(model);

}

public ActionResult Logoff()

{

FormsAuthentication.SignOut();

return RedirectToAction("Index", "Home");

}

}

}

**HomeController.cs:**

namespace Student.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

return View("Index");

}

[Authorize(Roles ="admin")]

public ActionResult About()

{

ViewBag.Message = "Your application description page.";

return View();

}

public ActionResult Contact()

{

ViewBag.Message = "Your contact page.";

return View();

}

}

}

**ManageController.cs:**

using Student.Models;

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace Student.Controllers

{

public class ManageController : Controller

{

UserContext userContext = new UserContext();

LectureContext lectureContext = new LectureContext();

TestContext testContext = new TestContext();

// GET: Manage

public ActionResult Index()

{

return View(userContext.GetUser(User.Identity.Name));

}

public ActionResult LectureList()

{

return View(lectureContext.Lecture);

}

public ActionResult CreateLecture()

{

return View();

}

[HttpPost]

public ActionResult CreateLecture(Lecture lecture)

{

lecture.date = DateTime.Now;

lectureContext.Lecture.Add(lecture);

lectureContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult DeleteLecture(int id)

{

Lecture lecture = lectureContext.Lecture.Find(id);

return View("DeleteLecture", lecture);

}

[HttpPost]

public ActionResult DeleteLecture(int id, FormCollection collection)

{

Lecture lecture = lectureContext.Lecture.Find(id);

lectureContext.Lecture.Remove(lecture);

lectureContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult EditLecture(int id)

{

Lecture lecture = lectureContext.Lecture.Find(id);

return View("EditLecture", lecture);

}

[HttpPost]

public ActionResult EditLecture(Lecture lecture)

{

lecture.date = DateTime.Now;

lectureContext.Entry(lecture).State = EntityState.Modified;

lectureContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult TestList()

{

return View(testContext.Test);

}

public ActionResult CreateTest()

{

return View();

}

[HttpPost]

public ActionResult CreateTest(Test test)

{

testContext.Test.Add(test);

testContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult GoTest(int id)

{

Test test = testContext.Test.Find(id);

return View(test);

}

[HttpPost]

public ActionResult GoTest(Test test, string names)

{

if (test.response == names)

{

int userId = userContext.GetUser(User.Identity.Name).Id;

Mark mark = new Mark();

mark.mark = 5;

mark.Testid = test.id;

mark.Userid = userId;

testContext.Mark.Add(mark);

testContext.SaveChanges();

}

else

{

int userId = userContext.GetUser(User.Identity.Name).Id;

Mark mark = new Mark();

mark.mark = 2;

mark.Testid = test.id;

mark.Userid = userId;

testContext.Mark.Add(mark);

testContext.SaveChanges();

}

return RedirectToAction("Index");

}

public ActionResult MarkList()

{

return View(testContext.Mark.Include(e => e.user));

}

public ActionResult DeleteTest(int id)

{

Test test = testContext.Test.Find(id);

return View("DeleteTest", test);

}

[HttpPost]

public ActionResult DeleteTest(int id, FormCollection collection)

{

Test test = testContext.Test.Find(id);

testContext.Test.Remove(test);

testContext.SaveChanges();

return RedirectToAction("Index");

}

public ActionResult UserList()

{

return View(userContext.User);

}

public ActionResult DeleteUser(int id)

{

User user = userContext.User.Find(id);

return View("DeleteUser", user);

}

[HttpPost]

public ActionResult DeleteUser(int id, FormCollection collection)

{

User user = userContext.User.Find(id);

foreach (Mark m in testContext.Mark)

{

if (id == m.Userid)

testContext.Mark.Remove(m);

}

testContext.SaveChanges();

userContext.User.Remove(user);

userContext.SaveChanges();

return RedirectToAction("Index");

}

}

}