Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ВЫСШАЯ ШКОЛА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

И ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Направление подготовки: 09.03.04 – Программная инженерия

Профиль: Технологии разработки информационных систем

КУРСОВАЯ РАБОТА

Форум

Студент 2 курса

Группы 11-707

«\_\_\_\_» \_\_\_\_\_\_\_ 2019 г. подпись Саитов Азат Уралович

Научный руководитель

\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_

уч. степень уч. звание

«\_\_\_\_» \_\_\_\_\_\_\_2019 г. подпись Катеев Арслан Дамирович

Казань - 2019г.

Оглавление

[Введение 3](#_Toc8600736)

[Глава I. Подготовка к разработке и обзор технологий. 5](#_Toc8600737)

[C# 7.3 6](#_Toc8600738)

[.NET Core 2.2 6](#_Toc8600739)

[Entity Framework Core 2.2 7](#_Toc8600740)

[MVC 7](#_Toc8600741)

[ASP.NET Core MVC 8](#_Toc8600742)

[JavaScript 9](#_Toc8600743)

[React 9](#_Toc8600744)

[PostgreSQL 11.2 10](#_Toc8600745)

[SCRUM 10](#_Toc8600746)

[Trello 10](#_Toc8600747)

[Git 11](#_Toc8600748)

[GitHub 11](#_Toc8600749)

[Техническое задание 12](#_Toc8600750)

[Глава II. Основные этапы разработки 13](#_Toc8600751)

[Заключение 16](#_Toc8600752)

[Список литературы 17](#_Toc8600753)

# Введение

В рамках учебного процесса, для получения практического опыта разработки приложений, студентам курса были выделены 4 проекта. В результате распределения, наша команда была назначена на разработку корпоративного проекта под названием «Форум».

Форум - платформа для общения между пользователями интернета.

Вся суть работы форумов заключается во взаимодействии пользователей друг с другом путём решения каких-либо вопросов, на которые они хотят получить ответы. Для этого в пределах форума создаются темы, где каждая тема затрагивает конкретную сферу общей тематики форума. В определённой теме пользователи создают топики со своими вопросами, а другие пользователи, в свою очередь, комментируют данные топики, таки образом отвечая на поставленный вопрос.

Полезность данного проекта заключается в том, что люди объединяются в решениях проблем, ко всему прочему, вопросы и ответы сохраняются в базе данных, и в дальнейшем могут быть полезны как участникам форума, так и любым пользователям сети интернет, которые ищут решение своей проблемы с помощью поисковых систем.

Актуальность данного проекта заключается в том, что некоторым современным реализациям в сфере IT требуется возможность создания нескольких форумов в пределах одного веб-сервиса, а также понятный и доступный интерфейс для пользователей.

Практическая задача, решаемая в рамках курсовой работы - реализация платформы для общения людей, позволяющая пользователям задавать интересующие их вопросы, получать на них ответы, а также проводить различного рода обсуждения.

На основе полученных на курсе разработки приложений на платформе .NET знаний и опыта, в данной курсовой будет рассмотрена разработка веб-приложения на основе языка C# и фреймворков .NET Core серверной и JavaScript-библиотеки React в качестве клиентской части. Для взаимодействия с базой данных была выбрана СУБД PostgreSQL.

Поскольку веб-приложение было спроектировано с использованием слоистой архитектуры, произошло разделение на проекты в серверной части с учетом распределения обязанностей и взаимодействия клиентской и серверной части приложения.

В рамках курсовой работы требуется разработать веб-приложение на основе технологии ASP.NET Core, а именно интернет-форум.

Также необходимо обеспечить интеграцию со сторонними проектами посредствам сетевой шины.

В процессе разработки необходимо использовать технологию доступа к данным Entity Framework Core, JavaScript библиотеку React, а также другие фреймворки, благоприятствующие качественной разработке проекта.

В итоге должно получится полноценное веб-приложение, с помощью которого пользователь сможет создавать топики, оставлять комментарии, читать обсуждения и общаться с другими пользователями.

# Глава I. Подготовка к разработке и обзор технологий.

Начало работы над данной курсовой началось со знакомства с принципами командной разработки и способами разработки проектов. Осуществлялся поиск информации о термине технического задания, как формируются подобные документы и для чего именно они нужны.

Для лучшего понимания способов осуществления корпоративной разработки были рассмотрены методы управления проектами. Ко всему прочему, были затронуты важность и принципы документирования проектов.

Далее в формате лекций студентам были рассказаны основополагающие принципы разработки проектов - основы проектирования, построения архитектуры и разработка на платформе .NET Core. Досконально были изучены паттерны проектирования веб-приложений, а также другие элементы, способные помочь разработке на .NET Core. Были получены знания о базах данных, их роли в веб-приложении и способах использования.

С целью лучшего понимания концепции разработки веб-приложений, множество лекционных занятий было посвящено разработкам клиентской части, в том числе, языку программирования JavaScript. Также, были изучены способы взаимодействия веб-приложений (приложений вида клиент-сервер), а конкретно, основы передачи информации с помощью протоколов HTTP и TCP/IP.

Для грамотной разработки и поддержания веб-приложений было изучено понятие рефакторинга (от англ. Refactoring - процесс изменения внутренней структуры программы, не затрагивающий её внешнего поведения и имеющий целью облегчить понимание её работы).

Помимо всего прочего полученные знания применялись на небольших практических заданиях, которые позволяли лучше усвоить материал, а также помочь тем, у кого возникали определённые проблемы на этапе получения информации или выполнения практического задания.

Для решения поставленной перед нами задачи, было необходимо изучить следующие технологии:

## C# 7.3

C# – объектно-ориентированный язык программирования, созданный в 2000 году группой инженеров компании Microsoft специально для  платформы Microsoft .NET Framework и относящийся к семейству языков с С-подобным синтаксисом, таких как C++, Java, а также Objective-C.

C# содержит в себе ключевые паттерны популярных языков программирования, такие как: статическая типизация, анонимные функции с поддержкой замыкания, язык запросов для работы с коллекциями LINQ, делегаты, атрибуты, события, полиморфизм, перегрузку операторов, исключения, итераторы, а также обобщенные типы и методы.

## .NET Core 2.2

.NET Core – это бесплатная, кроссплатформенная универсальная платформа для разработки с открытым исходным кодом, выпущенная в 2016 году компанией Microsoft. .NET Core основана на платформе .NET Framework, .NET Core отличается от нее модульностью, и возможностью применять облачные технологии. Модульность .NET Core - позволяет обновлять каждый ее отдельный компонент через менеджер пакетов NuGet, в то время как, в .NET Framework все модули обновляются целиком.

Еще одним отличием .NET Core от .NET Framework является разделение между библиотекой CoreFX и средой выполнения CoreCLR. CoreFX - это набор библиотек базовых классов .NET Core, составляющих пространства имен System.\*. CoreCLR - среда выполнения, которая содержит в себе компилятор, встроенный сборщик мусора, а также множество других компонентов.

.NET Core 2.2 – это последняя версия платформы .NET Core, вышедшая в 2018 году при поддержке корпорации Microsoft. Свежий релиз .NET Core содержит в себе улучшенную диагностику в рантайме, поддержку микропроцессорного ядра ARM32, и поддержку облачной службы управления удостоверениями Azure Active Directory для SQL Client.

## Entity Framework Core 2.2

Entity Framework Core – это расширяемая, кроссплатформенная версия объектно-ориентированной технологии доступа к данным Entity Framework. Entity Framework Core работает по принципу object-relational mapping - отображение данных на реальные объекты. Другими словами, Entity Framework Core содержит в себе более высокий уровень абстракции, что позволяет взаимодействовать с базой данных напрямую через объекты, не затрагивая ее таблицы, индексы, первичные и внешние ключи.

Так ключевым понятием в Entity Framework Core является понятие сущности или entity, entity определяет набор свойств, которые связаны с объектом.

Также поскольку Entity Framework Core основан на технологии ORM это позволяет использовать язык запросов LINQ для выборки данных из базы данных.

LINQ или Language-Integrated Query – представляет из себя язык запросов к источнику данных. Источником данных может являться объект, реализующий интерфейс IEnumerable (например, стандартные коллекции, массивы), набор данных DataSet, документ XML. Но вне зависимости от типа источника LINQ позволяет применить ко всем один и тот же подход для выборки данных.

## MVC

MVC – Model-View-Controller или Модель-Представление-Контроллер – это архитектурный шаблон, который описывает схему разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента практически не затрагивает остальные компоненты.

Модель – представляет собой данные и правила, которые используются для работы с данными, которые представляют концепцию управления приложением. Модель даёт контроллеру представление данных, которые запросил пользователь (сообщение, страницу книги, фотоальбом, и тому подобное). Модель данных будет одинаковой, вне зависимости от того, как мы хотим представлять их пользователю. Поэтому мы выбираем любой доступный вид для отображения данных.

Представление – обеспечивает различные способы представления данных, которые получены из модели. Он может быть шаблоном, который заполняется данными. Может быть несколько различных видов, и контроллер выбирает, какой подходит наилучшим образом для текущей ситуации.

Контроллер – управляет запросами пользователя (получаемые в виде запросов HTTP GET или POST, когда пользователь нажимает на элементы интерфейса для выполнения различных действий). Его основная функция — вызывать и координировать действие необходимых ресурсов и объектов, нужных для выполнения действий, задаваемых пользователем. Обычно контроллер вызывает соответствующую модель для задачи и выбирает подходящий вид.

## ASP.NET Core MVC

ASP.NET Core MVC – платформа, созданная для разработки веб-приложений и программных интерфейсов с помощью архитектурного паттерна Model-View-Controller или MVC.

В ASP.NET Core MVC за основу модели берется некоторый класс, который можно дополнительно описать атрибутами для валидации.

В качестве контроллера выступают классы, наследуемые от классов Controller или ControllerBase.

Представлением, или клиентской частью, может стать отдельное приложение на отдельном фреймворке или компоненты, созданные и обработанные с помощью двигателя представлений Razor (который предоставляет возможность обрабатывать код на языке C# внутри своих страниц).

## JavaScript

JavaScript – это мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

JavaScript обладает следующими архитектурными чертами:

1. динамическая типизация
2. слабая типизация
3. автоматическое управление памятью
4. прототипное программирование
5. функции как объекты первого класса.

JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам – функции как объекты первого класса, объекты как списки, анонимные функции, замыкания – все это придаёт языку дополнительную гибкость.

## React

React – это JavaScript библиотека, созданная компанией Facebook в 2012 году специально для разработки пользовательских интерфейсов. Эта библиотека не содержит в себе классическое понимание контроллеров, моделей и шаблонов, вместо этого React использует компонентный подход. Из компонентов собирается весь пользовательский интерфейс.

React предоставляет язык шаблонов и некоторые callback-функции для отрисовки HTML. Весь результат работы React — это HTML.

Компонент React – это участок кода, который представляет часть веб-страницы. Каждый компонент – это JavaScript-функция, которая возвращает кусок кода, представляющего фрагмент страницы. Также компоненты занимаются тем, что хранят свое внутреннее состояние в памяти, например, какая закладка выбрана.

Для формирования страницы мы вызываем эти функции в определённом порядке, собираем вместе результаты вызовов и показываем их пользователю.

## PostgreSQL 11.2

PostgreSQL 11.2 – свободная объектно-реляционная система управлениями базами данных, выпущенная в 1995 году при поддержке Калифорнийского Университета в Беркли.

PostgreSQL базируется на языке [SQL](https://ru.wikipedia.org/wiki/SQL) и поддерживает многие из возможностей стандарта SQL:2011.

## SCRUM

Чтобы своевременно выполнять план и эффективно распределять время, в качестве метода управления разработкой был выбран SCRUM.

SCRUM – одна из наиболее популярных «методологий» разработки, содержащая набор принципов, правил, которые позволяют в короткие по времени итерации(спринты), разработать для конечного пользователя функционирующий продукт. На этапе планирования перед каждым спринтом определяются возможности к реализации, которые не меняются на всём его протяжении. При этом фиксированная длина каждого спринта обеспечивает процессу предсказуемость, а небольшая длительность – гибкость.

## Trello

Trello - это программное обеспечение для управление проектами в небольших командах, разработанное в 2011 году компанией Fog Creek Software.

Для организации задач используется доска с карточками. Trello ограничил поддержку тегов в виде десяти цветных меток, которые можно переименовать. Карточки поддерживают комментарии, вложения, сроки выполнения и контрольные списки. Как правило, задачи делятся на три типа:

1. Запланированные
2. Текущие
3. Выполненные

Изначально все задачи помещаются в первый список, затем пока они находятся в процессе – во втором, после выполнения законченная задача помещается в третий список.

## Git

Git - система контроля версий, дающая разработчикам возможность отслеживать изменения в файлах и работать совместно с другими разработчиками. Git известен своей скоростью, простым дизайном, поддержкой нелинейной разработки, полной децентрализацией и возможностью эффективно работать с большими проектами.

Git хранит данные в виде своеобразных снимков. Каждый раз, когда вы сохраняете состояние своего проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок.

## GitHub

GitHub - веб-сервис, предназначенный для хранения информации на удаленном сервере, а также командной разработки различных проектов. Данный сервис онлайн-хостинга репозиториев, обладающий всеми функциями распределённого контроля версий и функциональностью управления исходным кодом, реализует все возможности Git и даже больше. Обычно он используется вместе с Git и даёт разработчикам возможность сохранять их код онлайн, а затем взаимодействовать с другими разработчиками в разных проектах.

## Техническое задание

Техническое задание – это документ, содержащий требования и условия выполнения задания, предоставляемый заказчиком. Данный документ содержит в себе цель разрабатываемого проекта и его основные характеристики. Помимо всего вышеописанного техническое задание устанавливает показатели качества и технико-экономические требования, определяет порядок и условия работ.

Для рационализации процесса корпоративной разработки, специально для проекта «Форум» было составлено техническое задание. Готовое веб-приложение должно обладать следующими возможностями:

* Возможность аутентификации пользователя
* Возможность создания системы прав доступа
* Возможность создания новых форумов через учётную систему
* Возможность создания новых тем через учётную систему
* Возможность создания новых сообщений в теме
* Возможность создания приватных форумов, тем
* Возможность предоставления прав пользователям на приватные форумы, темы
* Возможность установки личных данных
* Возможность установки html подписи
* Возможность использования emoji/смайлов/стикеров
* Возможность общего чата внутри форума
* Возможность встраивания чата фрейма

# Глава II. Основные этапы разработки

На основе изученного материала началась разработка платформы для общения пользователей под названием «Форум». Как уже говорилось ранее, архитектура проекта была построена многослойно - в зависимости от функций общий проект был разделен на 3 проекта:

* Forum.Core отвечает за модели и конфигурацию общего проекта
* Forum.Infrastructure отвечает за инфраструктуру и работу с базой данных
* Forum.Web который содержит клиентскую часть веб-приложения

После распределения обязанностей, началась разработка. Задачу каждого члена команды можно было узнать с помощью доски Trello, где все было распределено лидером команды.

Задача заключалась в разработке доменной модели(сущности) для форума, логики взаимодействия с базой данных и контроллеров.

Разработка начинается с проектирования доменной модели, поэтому с другим членом команды первым делом были спроектированы базовые сущности, которые находятся в проекте Forum.Core:

* Identity – абстрактный класс, который наследуют все нижеследующие классы. Содержит в себе свойство:

1. Id (уникальный идентификатор объекта, который наследует абстрактный класс Identity)

* Forum – класс, описывающий форум. Содержит в себе свойства:

1. Name (название форума);
2. Description (краткое описание форума).

* Theme – класс, описывающий тему. Содержит в себе свойства:

1. Name (название темы);
2. Description (краткое описание темы);
3. ForumId (уникальный идентификатор форума, которому принадлежит данная тема).

* Topic – класс описывающий вопрос(обсуждение). Содержит в себе свойства:

1. Head (название топика);
2. Text (описание топика);
3. CreatorId (уникальный идентификатор пользователя, написавшего топик);
4. CreationDate (дата создания топика);
5. ThemeId (уникальный идентификатор темы, которой принадлежит этот топик).

* Comment – класс, описывающий комментарий к топику. Содержит в себе свойства:

1. WriterId (уникальный идентификатор пользователя, который написал комментарий);
2. Text (текст);
3. CreationDate (дата создания комментария);
4. TopicId (уникальный идентификатор топика, которому принадлежит этот комментарий).

* User – класс, описывающий пользователя. Содержит в себе свойства:

1. Nickname (ник пользователя);
2. Email (email пользователя);
3. Password (пароль от аккаунта пользователя);
4. RegistrationDate (дата регистрации пользователя).

Поскольку работа выполнялась удалённо, с помощью веб-сервиса GitHub разработка происходила весьма продуктивно, так как все использовали актуальные версии проекта, а последующие изменения быстро фиксировались с помощью GitHub, после чего каждый член команды снова обновлял локальную версию проекта до новой.

Затем была настроена работа с базой данных и созданы репозитории – адаптеры для работы с объектами в базе данных в проекте Forum.Infrastructure. В процессе были созданы классы:

* ForumDbContext – класс, наследующийся от Microsoft.EntityFrameworkCore.DbContext, который отвечает за прямую связь с таблицами для обозначенных сущностей, таких как Forum, Theme.
* ForumRepository – репозиторий для операций получения, создания и удаления топиков. Наследуется от интерфейса IForumRepository.
* ThemeRepository – репозиторий для операций получения, создания, изменения и удаления комментариев. Наследуется от IThemeRepository.
* IForumRepository – интерфейс для ForumRepository, определяющий, какие методы будут у данного репозитория.
* IThemeRepository – интерфейс для ThemeRepository, определяющий, какие методы будут у данного репозитория.

Для Forum.Web были установлены зависимости между проектами, реализации, и необходимые настройки для DI контейнера, для того, чтобы исключить прямые зависимости между классами. Также, был внедрен в этот проект [ASP.NET](http://asp.net/) MVC – стандартная реализация паттерна проектирования Model-View-Controller и созданы контроллеры - системные элементы для переадресации HTTP запросов в системы моего проекта:

* ForumController – контроллер для обработки HTTP запросов, связанных с сущностью Forum, были реализованы стандартные, базовые функции CRUD (create, read, update, delete) для взаимодействия с хранилищем через репозиторий ForumRepository.
* ThemeController - контроллер для обработки HTTP запросов связанных с сущностью Comment, были реализованы стандартные базовые функции CRUD (create, read, update, delete) для взаимодействия с хранилищем через ThemeRepository.

# Заключение

В рамках данной курсовой работы я получил новые знания, а также практические умения, связанные с корпоративной разработкой веб-приложений. Работа подразделялась на 5 этапов:

1. Проектирование
2. Принятие определенной структуры проекта
3. Разделение обязанностей
4. Разработка
5. Тестирование

В результате выполнения данной курсовой работы было получено веб-приложение под названием «Форум», с помощью которого можно создавать множество форумов, осуществлять общение на его просторах путём создания новых тем, а в темах, в свою очередь, топиков, а также с возможностью комментировать данные топики.

В рамках работы над этой курсовой работой были изучены базовые принципы разработки, построения веб-приложений и их архитектура, некоторые паттерны проектирования, а также методы управления проектами. Также были изучены принципы работы с [ASP.NET](http://ASP.NET) MVC, построение приложения с помощью MVC. После выполнения каждого этапа нашей команде удалось получить работающее веб-приложение с возможностью общения между пользователями.

# Список литературы

* Джон Скит, C# для профессионалов
* Джозеф Альбахари, C# in a Nutshell
* Хабр <https://habr.com/>
* Документы Microsoft <https://docs.microsoft.com/en-us/dotnet/core/>
* GitHub, Документация по .NET Core, <https://github.com/dotnet/core>
* MSDN, Обзор .NET 2015 <https://channel9.msdn.com/Events/Visual-Studio/Connect-event-2015/NET-2015-Overview>
* MSDN, .NET goes cross-platform with .NET Core <https://msdn.microsoft.com/magazine/mt694084>
* Microsoft Developer Blog, Введение в .NET Core <https://devblogs.microsoft.com/dotnet/introducing-net-core/>
* Официальный сайт Microsoft, раздел .NET, ASP.NET MVC Pattern <https://dotnet.microsoft.com/apps/aspnet/mvc>
* Документация Microsoft, Обзор ASP.NET Core MVC <https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-2.2>
* Документация Microsoft, Обзор Entity Framework Core https://docs.microsoft.com/en-us/ef/core