Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Саратовский государственный технический университет

имени Гагарина Ю.А.»

Кафедра «Прикладные информационные технологии»

**ОТЧЕТ**

**о прохождении преддипломной практики**

Студента гр. Б2-ИФСТ-41

Синенковой Дианы Романовны

Принял: зав. кафедрой ПИТ, доцент

Долинина О.Н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_ 2019 г.

Саратов 2019

Оглавление

[Введение 3](#_Toc10631429)

[1. Описание предметной области 5](#_Toc10631430)

[1.1 Обзор существующих решений 6](#_Toc10631431)

[1.1.1 Обзор программного обеспечения «1С: Предприятие 8» 7](#_Toc10631432)

[1.1.2 Обзор программного обеспечения «SAP ERP» 9](#_Toc10631433)

[1.1.3 Обзор программного обеспечения «TECHCARD» 11](#_Toc10631434)

[1.1.4 Обзор программного обеспечения «ADEM-VX» 13](#_Toc10631435)

[1.1.5 Обзор программного обеспечения «CimatronE» 15](#_Toc10631436)

[1.2 Сравнительный анализ программного обеспечения 17](#_Toc10631437)

[2. Описание разрабатываемой информационной системы 19](#_Toc10631438)

[2.1 Функциональные возможности разрабатываемой информационной системы 19](#_Toc10631439)

[2.2 Состав информационной системы 28](#_Toc10631440)

[3. Практическая реализация информационной системы 37](#_Toc10631441)

[3.1 Выбор средств программной реализации 37](#_Toc10631442)

[3.1.1 Обзор платформы Microsoft.Net Framework 37](#_Toc10631443)

[3.1.2 Обзор языка программирования C# 40](#_Toc10631444)

[3.1.3 Обзор технологии ASP.Net MVC 41](#_Toc10631445)

[3.1.4 Обзор технологии ASP.NET Web API 43](#_Toc10631446)

[3.2 Выбор метода хранения данных 43](#_Toc10631447)

[3.2.1 Обзор Visual Studio SQL Server Data Tools 43](#_Toc10631448)

[3.2.2 Обзор Entity Framework 44](#_Toc10631449)

[3.3 Структура серверной части информационной системы 48](#_Toc10631450)

[Список литературы 64](#_Toc10631451)

# Введение

В современном мире при быстром росте темпов производства и конкуренции на экономическом рынке, одним из важных факторов является возможность идти в ногу со временем, а значит, получать ощутимый эффект от автоматизации производства. Так как использование автоматизированных систем в технических и организационных сферах предприятия ускоряет работу, повышает качество и снижает затраты, продажа продукции предприятия приносит больше прибыли.

При автоматизации производства актуальной задачей становится разработка оптимизированных технологических процессов, которые впоследствии позволяют создавать высокопроизводительные машины. Таким образом, автоматизация становится самым перспективным направлением развития производства.

Так как человек перестает принимать непосредственное участие в процессе производства, повышается точность в производстве, которая позволяет производить продукцию с наименьшим количеством бракованных изделий. Кроме этого, улучшаются условия труда работников, что положительно сказывается на их здоровье и работоспособности. Автоматизация дает наилучший результат при массовом производстве с трудоемким технологическим процессом [1].

Каждая отдельная отрасль требует индивидуального подхода в вопросе автоматизации. Для разработки автоматизированных систем нужен огромный опыт в сфере данного производства, данные экспертов, а также современные технологии в сфере оборудования и информационных технологий.

При построении автоматизированных систем нужно учитывать различные факторы производства, а также искать наиболее оптимальное решение поставленной задачи. Такие системы должны включать в себя все уровни производства, начиная от оборудования и заканчивая финансовыми отделами и отделами управления. При построении автоматизированных систем наиболее эффективными подходами будут те, которые наиболее полно охватывают бизнес-процесс [2].

Целью выпускной квалификационной работы является разработка серверной части информационной системы учета технологических возможностей механообрабатывающего оборудования.

# Описание предметной области

Современное развитие машиностроения представляет собой постоянный рост объемов производства при повышении сложности проектных работ. Эта тенденция произошла из-за следующих факторов:

1. Повышается сложность изделий и увеличивается количество их деталей, при условии, что качество изготовления должно быть значительно выше, появляются новые материалы и методы их обработки;
2. Появляется требование к оригинальности продукта, который сможет стать достойным конкурентом на рынке, при условии, что себестоимость продукта будет снижена;
3. Увеличивается количество оборудования с программным управлением, которое требует разработку дополнительного программного обеспечения и более точного проектирования операций;
4. Появляются требования к уменьшению сроков изготовления изделий и к наиболее быстрой подготовке производства к выпуску новых разработок в целях выживания в конкуренции рыночной экономики.

Таким образом, автоматизация технологического процесса производства основана на использовании всевозможных систем автоматического проектирования, баз данных, хранящих данные о процессах производства и его техническом обеспечении, и экспертных системах, помогающих в решении различных задач производства [3].

Для проектирования технологического процесса на механообрабатывающем предприятии используются следующие документы и данные:

1. чертеж детали,
2. чертеж заготовки,
3. программа производства,
4. полный перечень оборудования, которым располагает данное предприятие.

Результатом проектирования является план технологического процесса, но он решает только основные технологические задачи, такие как: последовательность операций, установление границ между одной и другой операцией, количество операций, определение установочных баз и места крепления деталей. Такой план не является исчерпывающим, он не раскрывает детали производства, такие как последовательность переходов, не рассчитываются численные значения точности и операционные припуски.

Этапами проектирования технологического процесса механической обработки являются:

1. Подготовка конструкторско-технологической информации;
2. Кодирование конструкторско-технологической информации;
3. Этап проектирования, который включает в себя проектирование маршрутов обработки, расчеты припусков и размеров, проектирование операций механической обработки, оценка времени, затраченного на изготовление детали, технологичности и экономического эффекта от изготовления [4].

# Обзор существующих решений

Для проектирования и разработки информационной системы следует проанализировать существующие решения, которые используются на современном производстве, оценить функционал и обратить особое внимание на недостатки.

## Обзор программного обеспечения «1С: Предприятие 8»

Одним из часто используемых на отечественном рынке аналогом является система программ «1С: Предприятие», пример вида окна системы представлен на Рисунке 1.1. Данное программное обеспечение состоит из основной платформы и прикладных решений, которые разработаны на ее базе. «1С: Предприятие» позволяет автоматизировать деятельность организаций и частных лиц. Конечные пользователи используют прикладные решения, а сама платформа выступает только базой для создания прикладных программ, она не используется конечными пользователями в чистом виде. Такой подход делает эту платформу наиболее подходящей для решения различных задач предприятия любой сферы деятельности [5].

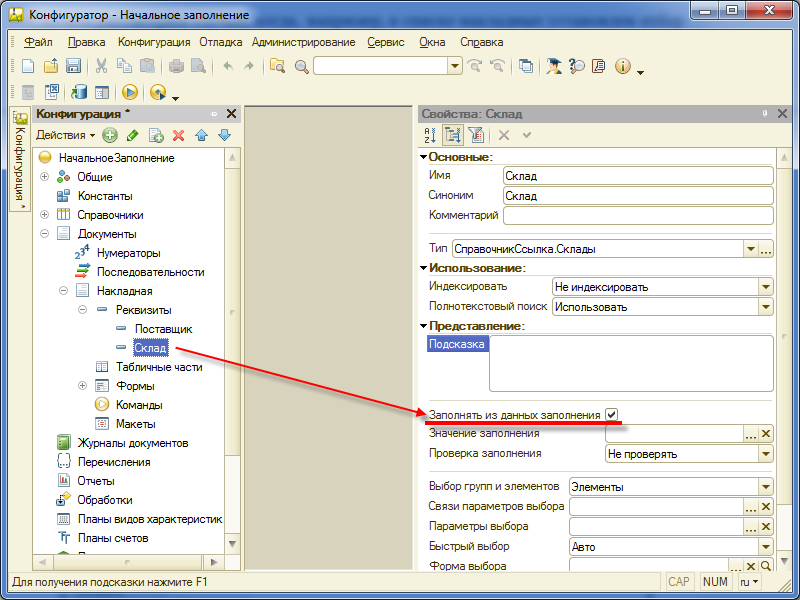


Рисунок 1.1 - Окно системы "1С: Предприятие 8"

Достоинства решения:

«1С: Предприятие 8» является универсальным решением для предприятий различных направлений и имеет широкий спектр функций:

* Автоматизация предприятий различного типа и сферы деятельности;
* Ведение бухгалтерского учета с несколькими счетами, различными измерениями учета, составление стандартизированных отчетов;
* Инструменты для управленческого учета, построения отчетности, основанной на аналитике;
* Инструменты для планирования бюджета;
* Составление смет, планирование закупок сырья, расчет заработной платы сотрудников, управление персоналом.

Дополнительным достоинством данного программного решения является возможность использовать платформу через сеть Интернет.

Помимо достоинств данного программного обеспечения при применении его для решения поставленной задачи были выявлены следующие недостатки:

1. Внедрение данного программного обеспечения в производственный процесс осуществляет партнер-внедренец, что является небезопасным решением, если предприятие является организацией закрытого типа, или штатный сотрудник, обученный разработке на данной платформе, что может привести к тратам материальных ресурсов и времени на обучение;
2. Адаптация программного обеспечения под конкретные нужды производства достаточно трудоемка и занимает большое количество времени и материальных ресурсов.

# Обзор программного обеспечения «SAP ERP»

SAP ERP – это пакет инструментов для планирования общеорганизационных ресурсов, вид окна приведен на Рисунке 1.2. Данный продукт является комплексным решением для управления поставками, жизненным циклом продукции, сотрудниками, финансами и взаимоотношениями с клиентами. Продукт может использоваться через сеть Интернет, по локальной сети, а также через Интернет и по локальной сети одновременно, благодаря специальной настройке. Компания имеет огромную сеть партнеров, которые предоставляют услуги по внедрению и настройке системы.

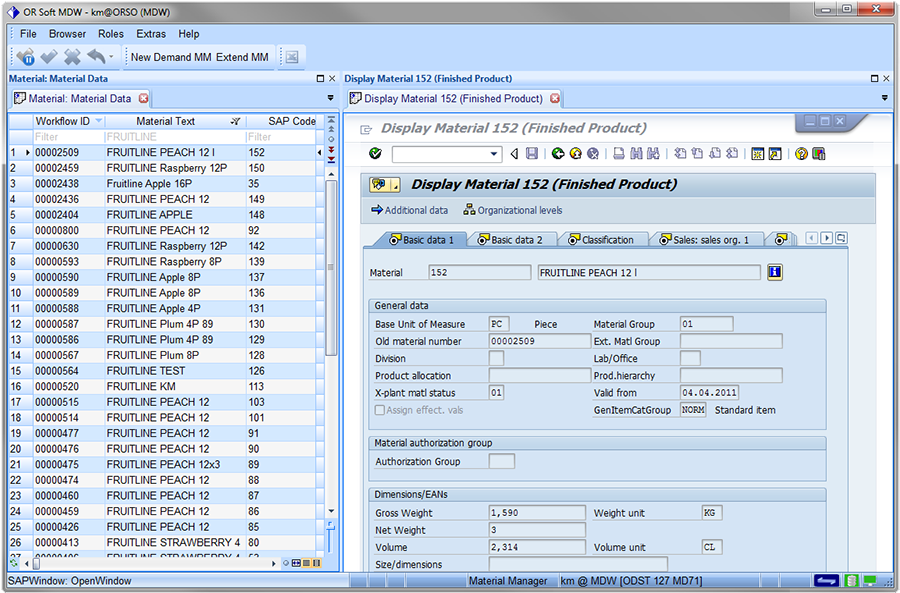


Рисунок 1.2 – Вид окна программного обеспечения SAP ERP

SAP ERP является одним из популярных продуктов на международном рынке ERP-систем, поэтому разработка программных модулей для данного продукта также является очень популярной. Программное обеспечение с набором нужных для предприятия программных модулей поможет значительно оптимизировать производственные процессы, тем самым повысив конкурентоспособность предприятия. Кроме этого, данное программное обеспечение позволяет автоматизировать производство [6].

Достоинства:

* Поставщик предоставляет техническую поддержку через сеть Интернет, благодаря которой пользователи получают доступ ко всем функциям и справкам в любой точке;
* Отслеживание полного документооборота предприятия, будь то планирование продукта или бухгалтерские отчеты;
* Легкое масштабирование производства благодаря гибкости и модульности системы;
* Интеграция с системами, которые ранее использовались на производстве;
* Обмен сведениями с партнерами;
* Сокращение затрат — складских, трудовых (учет/контроль персонала), на капстроительство и прочих;
* Ускорение циклов закупки и оборачиваемости по задолженностям;
* Упрощение работы бухгалтерии, прочих отделов;
* Полноценная локализация;
* Возможность объединять с решениями сторонних разработчиков;
* Возможность использования системы для мобильного бизнеса.

Также преимуществом модулей SAP является быстрая и постоянная обновляемость данных. Система позволяет предоставить доступ к актуальным данным всем сотрудникам в режиме реального времени благодаря высокой отказоустойчивости [7].

Недостатки:

Несмотря на обширный список достоинств, для решения поставленной задачи функциями данного программного продукта, выявлены следующие недостатки:

* Отсутствует модуль с функциями, которые необходимы для решения данной задачи;
* Разработка нового модуля под данную платформу требует особых навыков, большого количества затрат материальных ресурсов и времени.

Данные недостатки критичны для нашей задачи.

# 1.1.3 Обзор программного обеспечения «TECHCARD»

Techcard – это система для проектирования маршрутов, технологических процессов для нужд производства. Данное решение позволяет автоматизировать формирование комплекта технической документации, технологических ведомостей и других документов, используемых на производстве. Система является комплексом программ, которые могут использоваться как пакетом, так и отдельными независимыми модулями. Пакет Techcard включает в себя:

* Систему, позволяющую организованно хранить конструкторскую и технологическую документацию, обеспечивающую тем самым взаимодействие между конструкторским и технологическим отделами;
* Систему автоматизированного проектирования технологического процесса для производства, позволяющую сформировать полный пакет технологической документации;
* Систему автоматизированного рабочего места расцеховщика, предназначенное для создания маршрутов и сводных ведомостей;
* Систему проектирования и оформления операционных эскизов, используемых в технологических документах. К тому же, TECHCARD позволяет работать с эскизами, созданными в системах стороннего программного обеспечения, такого как AutoCAD, SolidWorks, и других.

Система TECHCARD имеет следующие достоинства:

* Возможность взаимодействия со сторонними системами управления базами данных, таких как Oracle и InterBase 5.6;
* Автоматическое формирование различных ведомостей на изделие, а также подетальные и сводные ведомости, расцеховочные маршруты, ведомости заготовок, нормы времени и материала;
* Отдельные рабочие места для проектирования расцеховочных маршрутов и расчета норм материала. В зависимости от входных и выходных материалов, расцеховочные маршруты корректируются, тем самым сокращая время на исправление ошибок и неточностей в маршруте;
* Позволяет просматривать и печатать комплекты документов, тем самым помогая отследить неточности и проконтролировать технологический процесс;
* Позволяет увидеть состав изделия на рабочем месте, настроить параметры рабочего места для выдачи документации на деталь или изделие;
* Обеспечивает связь с внешними системами, а также доступ к параметрам технологического процесса в базе данных;

Данное программное обеспечение имеет большой ряд решаемых задач, отличную поддержку государственных стандартов документации.

Недостатком системы является:

* Неполное разграничение ролей. Некоторые функции доступны либо всем пользователям системы, либо нужные функции недоступны некоторой группе пользователей, из-за чего теряется время на обеспечение производства документацией;
* Внедрение системы в производство требует большого количества временных и материальных затрат на обучение.

# Обзор программного обеспечения «ADEM-VX»

ADEM является системой сквозного проектирования. Благодаря набору различных, но связанных между собой инструментов обеспечивает полное объединение работы сотрудников по подготовке производства, тем самым позволяет сократить время на разработку и налаживание производственного процесса.

Система позволяет автоматизировать решение проектных, конструкторских и технологических задач. ADEM подходит для машиностроительной промышленности, используется в производстве авиационной, аэрокосмической, приборной продукции, а также электронного оборудования. Система помогает наладить проектирование и производство сложных механизмов, пресс-форм и штампов. Данное программное обеспечение подходит для работы специалистов по техническому дизайну, конструкторам, технологам и архитекторам.

Система позволяет решать следующие задачи: проектирование и моделирование изделий, оформление конструкторской документации, проектирование технологических процессов, оформление технологической документации, программирование станков с числовым программным управлением, хранение данных в архивах, обновление данных, работа с трудовым нормированием сотрудников, управление справочниками.

Система состоит из нескольких модулей, таких как PDM, CAD, CAPP, CAM, NTR, Assembly, i-Ris. Модуль PDM (Product Data Management – система управления данными об изделии) позволяет структурированно хранить документы и управлять данными изделия. Модуль CAD (Computer Aided Design – система автоматизированного проектирования [10]) предназначен для плоского и объемного моделирования, создания чертежей и другой конструкторской документации. Модуль CAPP (Computer-Aided Process Planning – автоматизированная система проектирования технологических процессов и оформления документации [11]) позволяет автоматизировать проектирование единичных и групповых технологических процессов, оформить ведомости деталей по всем технологическим операциям для машиностроения и приборостроения в соответствии с Единой системой технологической документации. Данный модуль содержит справочники оборудования, инструментов, материалов и оснастки, позволяет автоматизировать расчет трудовых норм и расход материалов. Модуль CAM (Computer-Aided Manufacturing – система автоматизированного производства [10]) помогает создавать управляющие программы для различных видов станков и систем с ЧПУ, в том числе многоканальное оборудование. Управляющая программа создается в соответствии с общим технологическим процессом изготовления детали. Модуль ADEM NTR (Нормирование и технологические расчеты) позволяет быстро и эффективно определить норму времени на изготовление изделия и отдельных его деталей в автоматизированном режиме. Подсистема ADEM Assembly предназначена для эффективного проведения работ по сборкам изделия. Справочная система i-Ris, входящая в комплект программ ADEM, поддерживает многопользовательский режим работы и масштабируемость системы. Технология Net Remoting предназначена для взаимодействия удаленных объектов [12].

Достоинствами данного программного решения являются:

1. Полное создание технической документации, начиная с проектирования детали и заканчивая готовыми маршрутными картами;
2. Комплексный подход к разработке технологического процесса, который охватывает различные сферы производства;
3. Модули являются полноценным программным обеспечением.

Стоит также упомянуть о недостатках данной системы:

1. Большая стоимость данной системы;
2. Для внедрения в производство и обучения сотрудников, понадобится большое количество материальных ресурсов и времени.

# Обзор программного обеспечения «CimatronE»

Решения CAD/CAM компании Cimatron охватывают весь процесс изготовления инструментов - от составления документации до проектирования, применения инженерных изменений и программирования ЧПУ. При переходе на данное программное обеспечение наработанная база импортируется, поэтому перевести производство на работу с CimatronE не потребует большого количества времени.

Cimatron предоставляет значительные преимущества для пресс-форм, штамповочных и производственных цехов, что позволяет производству быть более конкурентоспособным [13].

Данное программное обеспечение имеет ряд достоинств:

* Импорт данных из различных форматов без потери информации, возможность устранения неточностей, пробелов и совпадений;
* Сокращение затрат времени на создание документации благодаря заранее заготовленным шаблонам, автоматическому исправлению и возможности ручной проверки созданных документов;
* Сохранение точности в разработке сложных изделий, расчет допусков, шероховатостей при конструировании различных поверхностей, балок, отверстий, скосов. Данный подход позволяет максимально улучшить качество изделий и их сборки;
* Возможность создания программ для станков с ЧПУ, что значительно сокращает процесс разработки технологического процесса и сокращает сроки изготовления изделия;
* Автоматизация формирования чертежей и конструкторской документации в соответствии с принятыми стандартами, которая позволяет сократить время на создание документации для выпуска новых изделий;
* Единая интегрированная система для создания детали или изделия. Таким образом, данное программное обеспечение позволяет оптимизировать процесс проектирования, сокращая время и устраняя ошибки во внесении изменений во все этапы изготовления изделия, начиная от конструирования и заканчивая разработкой маршрута и операционной документации. Работа в CimatronE позволяет организовать коллективную работу благодаря строгому разграничению прав доступа для специалистов различных направлений;
* Несмотря на то, что данное программное обеспечение является зарубежной разработкой, оно полностью русифицировано, что позволяет использовать систему на отечественных предприятиях;
* Программное обеспечение позволяет разрабатывать дополнительные модули для решения специфичных задач предприятия.

Недостатками данной системы можно указать:

1. Затраты времени на проверки правильности составления чертежей и технологических процессов;
2. Техническая поддержка системы несколько затруднена из-за того, что CimatronE является разработкой зарубежной компании в отличие от ADEM-VX и Techcard;
3. Большие материальные и временные затраты на обучение сотрудников.

# Сравнительный анализ программного обеспечения

На основании вышеизложенной информации следует сравнить существующие программные обеспечения по критериям, приведенным в Таблице 1.1:

Таблица 1.1 – Сравнительный анализ существующего программного обеспечения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Программное решение/  критерий | 1С: Предприятие 8 | SAP ERP | Techcard | ADEM-VX | CimatronE |
| Наличие справочников | - | - | + | + | + |
| Разделение ролей | + | + | - | + | + |
| Построение документации | - | - | + | + | + |
| Архив для документации | + | + | + | + | + |
| Возможность построения отчетов | - | + | - | + | - |
| Журналирование действий пользователей | - | + | - | - | + |

Таким образом, можно сделать вывод, что целью выпускной квалификационной работы является разработка программного обеспечения, позволяющего хранить информацию в справочниках, создавать, редактировать и удалять данные из базы технологических возможностей, хранить созданную документацию, строить и хранить отчеты по учету технологических возможностей предприятия, вести журнал действий пользователей с данными.

# Описание разрабатываемой информационной системы

Информационная система предназначена для ввода информации о всех технологических возможностях конкретного предприятия, ввода информации о детали, на основании которой создается стандартизированный технологический документ – маршрутная карта. Для отслеживания наличия и полноты документации по тому или иному изделию создаются отчеты по базе технологических возможностей и базе деталей.

Целевой аудиторией данной системы являются технологи механообрабатывающих предприятий.

# Функциональные возможности разрабатываемой информационной системы

Система должна обеспечивать разграничение ролей для отделов механообрабатывающего предприятия, а также обеспечивать разграничение прав доступа к данным для сотрудников различных предприятий. Таким образом системе необходимы следующие типы пользователей:

1. Администратор системы;
2. Технолог базы технологических возможностей;
3. Технолог по составлению маршрутных карт.

2.2 Функциональные возможности системы в зависимости от уровня доступа

Доступ к данным, которые являются наработками данного предприятия имеют только сотрудники этого предприятия.

Для роли «Администратор системы» предусмотрены следующие функции:

* Авторизация в системе;
* Добавление и редактирование предприятий, которые используют систему;
* Добавление и редактирование пользователей системы;
* Добавление, редактирование и удаление записей в базе технологических возможностей;
* Добавление и редактирование данных в справочниках (справочник оборудования, справочник приспособлений, справочник технологических переходов, справочник обрабатываемых поверхностей);
* Добавление и редактирование данных по детали, оборудованию и заготовке;
* Просмотр базы созданных маршрутных карт;
* Доступ к серверному приложению, который позволяет:

1. Просматривать отчеты по базе технологических возможностей предприятия;
2. Просматривать отчеты по базе деталей, оборудования и заготовкам;
3. Просматривать отчеты по работе сотрудников того или иного предприятия;
4. Просматривать журналы действий с базой данных;
5. Составлять отчет по наличию деталей для определенного изделия;
6. Экспортировать отчеты в формат PDF.

Для роли «Технолог базы технологических возможностей» доступны следующие функции:

* Авторизация в системе;
* Добавление записи в базу данных с пометкой «на проверку»;
* Редактирование отклоненных записей, автором которых является авторизованный сотрудник.

Для роли «Технолог по составлению маршрутных карт» предусмотрены следующие функции:

* Авторизация в системе;
* Добавление данных о детали;
* Добавление данных об оборудовании;
* Добавление данных о заготовке;
* Просмотр каталога готовых маршрутных карт своего предприятия;
* Формирование маршрутной карты;
* Сохранение маршрутной карты в базу.

2.3 Функционал серверной части информационной системы

Серверная часть обеспечивает следующие функции:

* Осуществление авторизации пользователей;
* Получение запросы из клиентской части;
* Отправка ответов на запросы из клиентской части;
* Обработка запросов клиента;
* Формирование запросов к базе данных;
* Отправка запросов в базу данных;
* Журналирование действий с базой данных;
* Формирование отчетов по базе деталей, оборудования и заготовок;
* Формирование отчетов по базе технологических возможностей;
* Формирование отчетов по сотрудникам;
* Формирование отчетов по наличию деталей для определенного изделия;
* Просмотр журнала действий с базой данных.

2.4 Входные и выходные данные

Входные данные:

Данные о предприятии:

* Сотрудник, ответственный за предприятие;
* Номер филиала;
* Наименование предприятия;
* Адрес предприятия.

Данные о сотруднике:

* Роль сотрудника в системе;
* Предприятие, на котором работает сотрудник;
* Имя;
* Фамилия;
* Отчество;
* Логин;
* Пароль.

Данные о записи в базе технологических возможностей:

* Предприятие, которому принадлежит данная запись;
* Сотрудник, создавшую данную запись;
* Статус записи (может быть в одном из трех состояний: «на проверку», «проверена», «отклонена»);
* Код оборудования (из справочника оборудования);
* Код приспособления (из справочника приспособлений);
* Код технологического перехода (из справочника технологических переходов);
* Код поверхности (из справочника поверхностей);
* Другие поля.

Данные справочника оборудования:

* Модель оборудования;
* Описание оборудования.

Данные справочника приспособлений:

* Значение приспособления;
* Описание приспособления.

Данные справочника технологических переходов:

* Описание технологического перехода.

Данные справочника поверхностей:

* Описание поверхности.

Данные о заготовке:

* Сотрудник, добавивший запись;
* Предприятие, владеющее данной записью;
* Код типа заготовки;
* Масса заготовки;
* Описание заготовки.

Данные о базовой поверхности заготовки:

* Наименование заготовки;
* Тип элементарной поверхности;
* Ориентация поверхности;
* Номер поверхности на заготовке;
* Координаты нулевой точки X;
* Координаты нулевой точки Y;
* Координаты нулевой точки Z;
* Внутреннее напряжение X;
* Внутреннее напряжение Y;
* Внутреннее напряжение Z;
* Показатель шероховатости Ra;
* Показатель шероховатости Rmax;
* Показатель шероховатости S;
* Показатель шероховатости tp;
* Показатель твердости HRC;
* Глубина наклепанного слоя;
* Возможность использования в качестве конструкторской базы.

Данные о детали:

* Сотрудник, добавивший запись о детали;
* Предприятие, владеющее записью о детали;
* Наименование детали;
* Марка материала;
* Объем производственной партии;
* Объем передаточной партии;
* Возможность изготовления центровых технологических отверстий;
* Примечание.

Данные о группах оборудования:

* Наименование группы;
* Примечание.

Данные о станке:

* Группа оборудования;
* Наименование станка;
* Код станка;
* Примечание.

Данные об изделии:

* Код изделия
* Детали, входящие в изделие.

Выходные данные:

Данные о маршрутной карте:

* Наименование маршрутной карты;
* Сотрудник, создавший карту;
* Дата создания.

Данные отчета о базе технологических возможностей:

* Предприятие, которому принадлежит данная запись;
* Сотрудник, добавивший запись;
* Статус записи;
* Код оборудования;
* Код технологического перехода;
* Код обрабатываемой поверхности.

Данные отчета о деталях:

* Предприятие;
* Сотрудник;
* Наименование детали;
* Марка материала;
* Объем производственной партии;
* Объем передаточной партии;
* Возможность изготовления центровых технологических отверстий;
* Примечание.

Данные отчета о заготовке:

* Предприятие, владеющее данной записью;
* Сотрудник, добавивший запись;
* Код типа заготовки;
* Масса заготовки;
* Описание заготовки.

Данные отчета о готовности изделия:

* Предприятие, владеющее данной записью;
* Сотрудник, описавший данное изделие;
* Перечень деталей, которые описаны в базе;
* Перечень деталей, которые не описаны в базе.

# Состав информационной системы

Информационная система включает в себя:

* блок авторизации пользователя;
* блок добавления записи в систему;
* блок импорта документа в систему;
* блок выбора, добавления и изменения справочников;
* блок проверки записи в базе;
* блок разделения пользователей по полномочиям на роли;
* блок журналирования процессов системы;
* блок добавления и редактирования детали;
* блок формирования маршрутной карты;
* блок вывода созданных ранее маршрутных карт;
* блок создания отчетов по информации, хранящейся в базе данных программного обеспечения.

Для уточнения предметной области можно использовать методологию функционального моделирования IDEF0 и DFD (диаграммы потоков данных).

Диаграмма IDEF0

IDEF0 – методология функционального моделирования использует графическую нотацию, предназначенную для описания бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность [16].

Стандарт IDEF0 представляет организацию как набор модулей, здесь существует правило — наиболее важная функция находится в верхнем левом углу, кроме того есть правило стороны:

* стрелка входа приходит всегда в левую кромку активности,
* стрелка управления — в верхнюю кромку,
* стрелка механизма — нижняя кромка,
* стрелка выхода — правая кромка.

Главной функцией системы для администратора является возможность отслеживать состояние системы и состояние данных, находящихся в базе данных. Поэтому для роли администратора модель IDEF0 будет выглядеть, как показано на Рисунке 2.1:

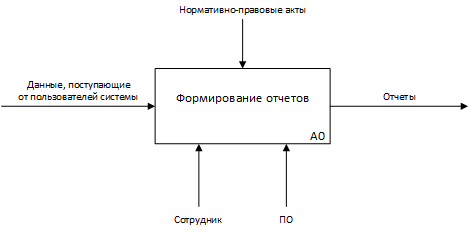


Рисунок 2.1 – Контекстная диаграмма IDEF0. Точка зрения: администратор системы

С точки зрения технолога по деталям главной функцией системы являет формирование маршрутной карты. Таким образом, для данной роли модель IDEF0 будет выглядеть так, как показано на Рисунке 2.2:

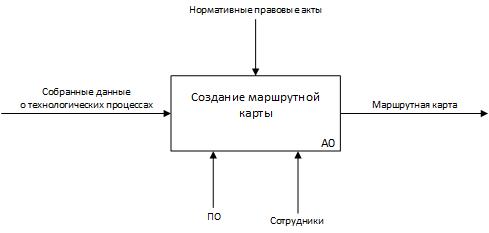


Рисунок 2.2 – Контекстная диаграмма IDEF0. Точка зрения: технолог по деталям

Для уточнения сложного процесса, следует разбить его на более простые подпроцессы. Для этого стоит произвести декомпозицию контекстных диаграмм.

Таким образом, для администратора декомпозиция диаграммы IDEF0 будет выглядеть, как показано на Рисунке 2.3:

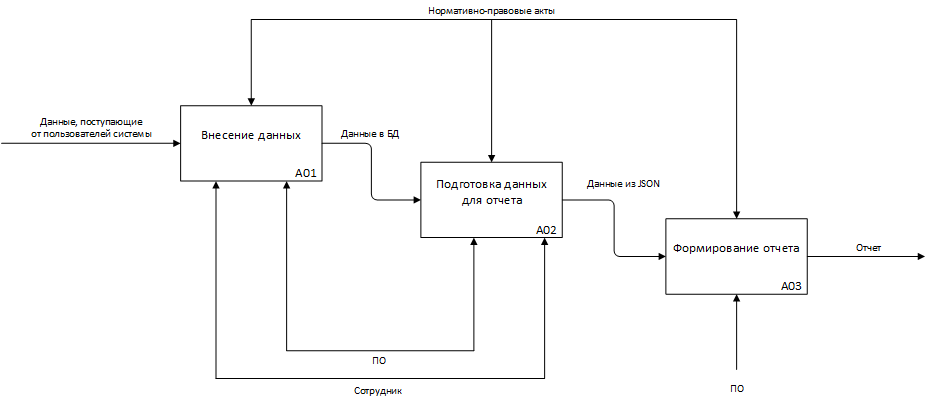


Рисунок 2.3 – Декомпозиция диаграммы IDEF0. Точка зрения: администратор системы

Для роли «Технолог по деталям» декомпозиция диаграммы IDEF0 будет выглядеть, как показано на Рисунке 2.4:

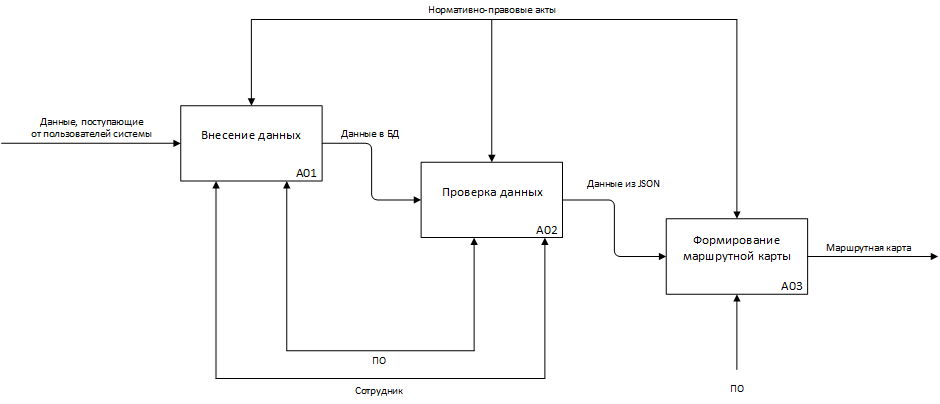


Рисунок 2.4 – Декомпозиция IDEF0. Точка зрения: технолог по деталям

Для того, чтобы проследить последовательность событий, синхронность и асинхронность событий, используется диаграмма IDEF3 [17]. Данная диаграмма представлена с точки зрения администратора на Рисунке 2.5:

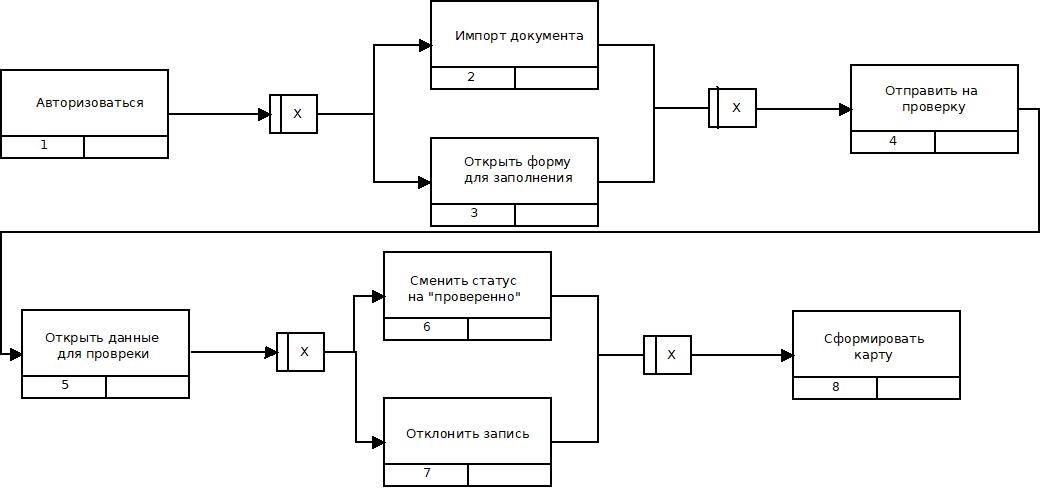
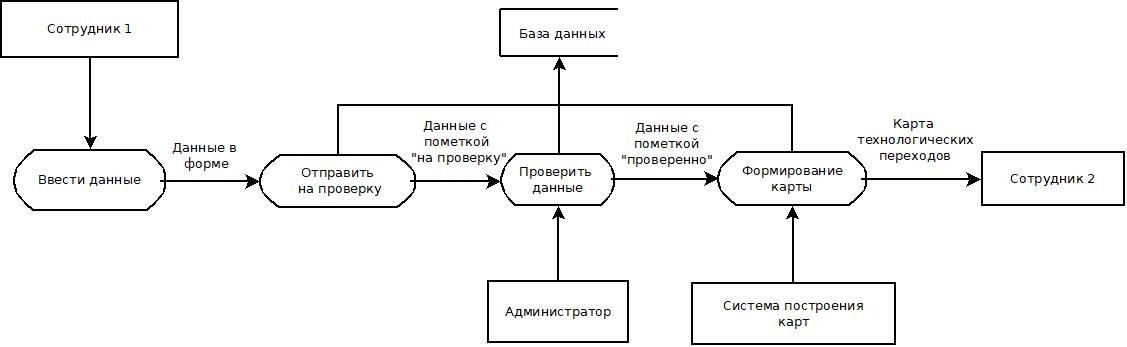


Рисунок 2.5 – Диаграмма IDEF3 с точки зрения администратора системы

Диаграммы потоков данных (DFD) являются основным средством моделирования функциональных требований проектируемой системы. С их помощью требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных. Главная цель таких средств – продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

Нотации DFD – удобное средство для формирования контекстной диаграммы, то есть диаграммы, показывающих их разрабатываемую БД в коммуникации с внешней средой. Одним из преимуществ нотации является возможность отображения на диаграмме места бизнес-процесса, в которых хранится информация, либо материальные ресурсы [18].

Модель предметной области, разработанная на основе методологий DFD, представлена на Рисунке 2.6:

Рисунок 2.6 – Диаграмма DFD

2.3 Использование объектно-ориентированных методологий для построения схем информационных процессов системы

Для уточнения описания предметной области строятся модели на основе методологии объектного анализа Unified Modeling Language (UML)– унифицированного языка моделирования.

В результате моделирования формируется уточненное описание предметной области. Использование языка UML для моделирования позволяет сделать модель понятной для всех участников проекта.

Прежде чем начать разработку автоматизированной системы, необходимо на уровне моделей детально описать предметную область, для которой эта система разрабатывается. Язык UML позволяет построить модели, дополняющие описание предметной области [19].

Диаграмма объектов предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.

Диаграммы объектов представляют статический вид системы с точки зрения проектирования и процессов, являясь основой для сценариев, описываемых диаграммами взаимодействия. Говоря другими словами, диаграмма объектов используется для пояснения и детализации диаграмм взаимодействия, например, диаграмм последовательностей [20]. Диаграмма объектов системы представлена на Рисунке 2.7.

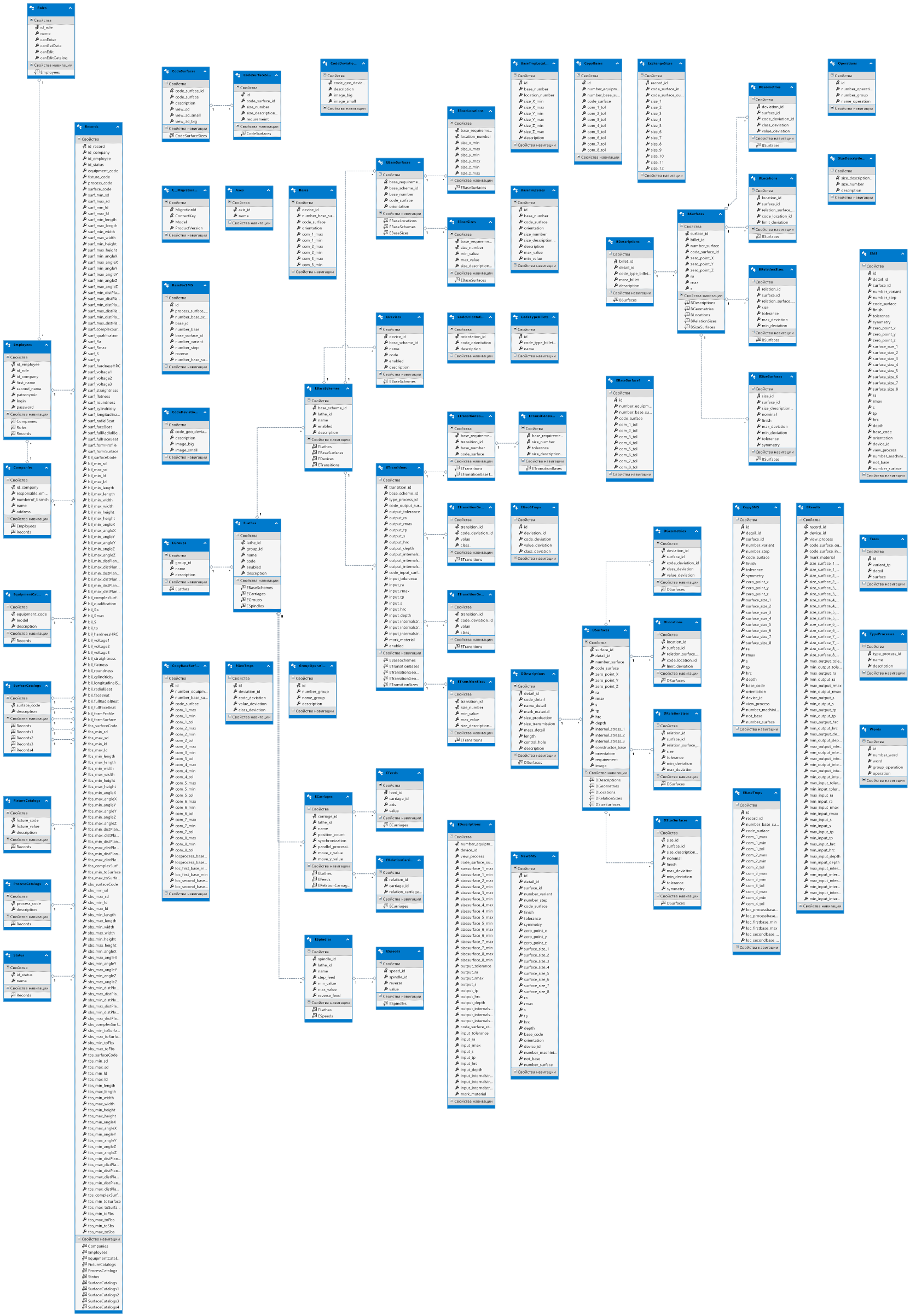


Рисунок 2.7 – Диаграмма объектов

Диаграмма прецедентов – это диаграмма, отражающая отношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Прецедент — возможность моделируемой системы (часть её функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат. Прецедент соответствует отдельному сервису системы, определяет один из вариантов её использования и описывает типичный способ взаимодействия пользователя с системой [21]. Диаграмма прецедентов представлена на Рисунке 9:

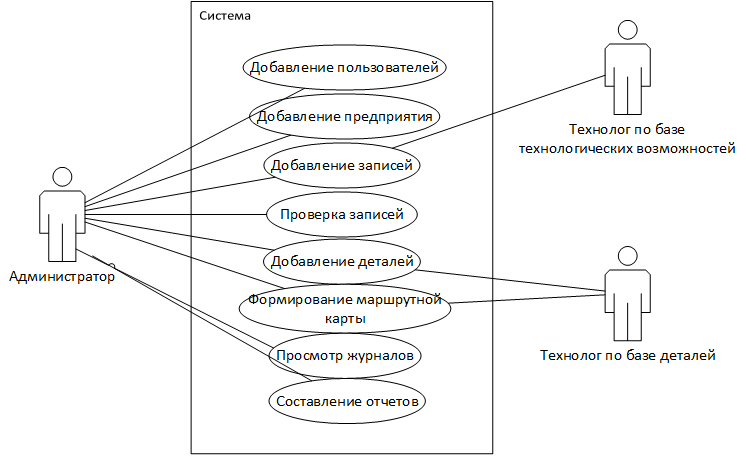


Рисунок 2.8 – Диаграмма прецедентов

Диаграмма взаимодействия - диаграмма, на которой показано взаимодействие объектов (обмен между ними сигналами и сообщениями), упорядоченное по времени, с отражением продолжительности обработки и последовательности их проявления [22]. Диаграмма взаимодействия представлена на Рисунке 2.9:

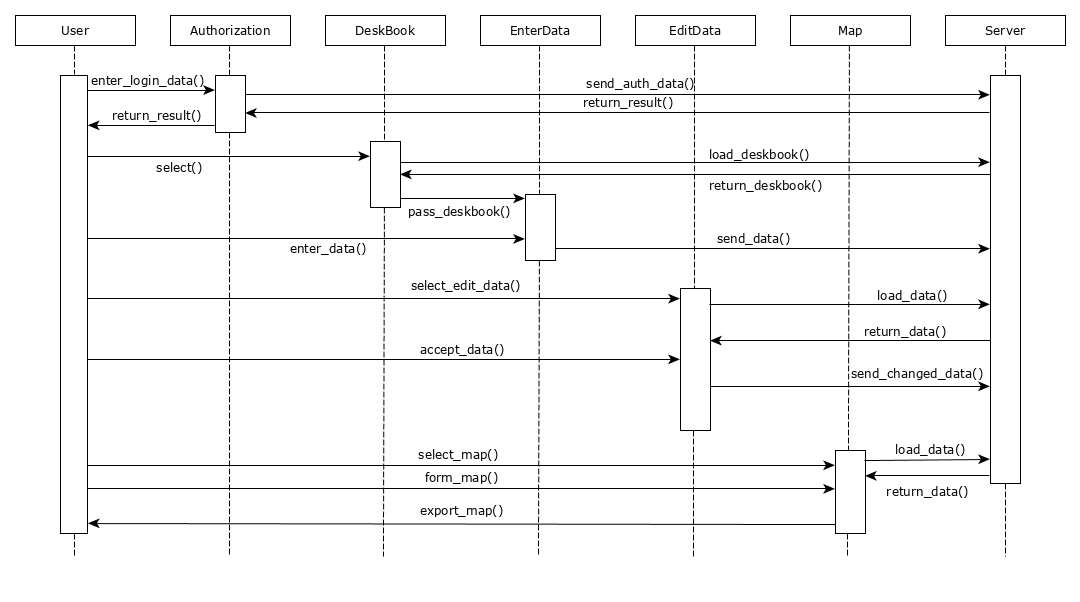


Рисунок 2.9 – Диаграмма последовательности

# Практическая реализация информационной системы

После этапа моделирования системы и описания ее с помощью различных методологий и диаграмм, идет этап практической реализации информационной системы.

# 3.1 Выбор средств программной реализации

Для реализации информационной системы учета технологических возможностей механообрабатывающего предприятия была выбрана платформа Microsoft.Net, а также язык программирования C#. Для создания базы данных используется технология Entity Framework, а для реализации приложения – ASP.Net Web Pages. Для передачи данных клиентскому приложению применяется технология Web API. Данный выбор обусловлен тем, что в основном для работы с информационной системой целевая аудитория будет использовать операционную систему Windows.

## Обзор платформы Microsoft.Net Framework

Платформа Microsoft .NET – это платформа для запуска программ, изначально разработанная для работы под операционную систему Windows. В отличие от набравшей популярность платформы Java, Microsoft .NET имеет совершенно иной принцип работы. Java создана как единая платформа для использования на различных устройствах и различных операционных системах.

Microsoft .NET имела совершенно иной смысл. Данная платформа позволяла писать на любом языке программирования под определенную операционную систему - Windows. Данный принцип достигался за счет лежащей в основе платформы общеязыковой среды исполнения – Common Language Runtime.

Однако из-за тенденций «одного приложения на всех устройствах» .NET начала плавно изменяться в сторону принципа «множество языков для множества систем». Так появилась улучшенная версия Microsoft .NET Framework - .NET Core, поддерживающая мультиплатформенную разработку приложений.

Рассмотрим подробнее структуру платформы. Она состоит из трех уровней:

1. Операционные системы;
2. Ядро платформы, состоящее из серверов .NET, подсистемы .NET Framework и набора различных сервисов .NET, позволяющих сократить время и повысить качество разработки приложения;

3. Общая среда разработки Visual Studio .NET. Данная среда обладает широким спектром инструментов для разработки на различных языках программирования, таких как: C#, F#, VisualBasic.NET, JScript.NET, C++ [23].

Структура данной платформы представлена на Рисунке 3.1:

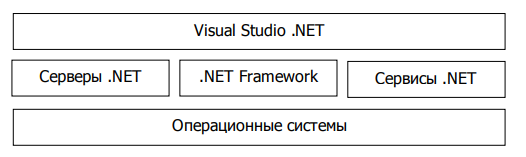


Рисунок 3.1 – Платформа Microsoft.Net

Рассмотрим детальнее подсистему Microsoft .NET Framework. Данная система обеспечивает исполнение .NET приложений. В ее состав входят общеязыковая среда исполнения (CLR) и библиотеки классов.

Основные возможности CLR:

* Управление производительностью и оптимизация;
* Автоматическая сборка мусора;
* Легкое использование компонентов программы, написанных на различных языках программирования;
* Обеспечение многопоточности;
* Автоматическая упаковка приложений и создание установочного приложения [25];
* И многое другое.

Ключевой функцией CLR является поддержка мультиязычности. Она достигается за счет перевода кода на Microsoft Intermediate Language (MSIL), что, по сути, является языком более низкого уровня из семейства Ассемблер [24].

Помимо CLR в базовый набор .NET Framework входит библиотека классов FCL.

Библиотека классов платформы (Framework Class Library) – это библиотека, содержащая стандартный набор пространств имен. Она общая для всех языков программирования, которые используются на платформе Microsoft .NET [26].

Библиотека классов включает в себя классы, которые позволяют реализовать следующие функции:

* Ввод и вывод данных, файловый и консольный;
* Реализация многопоточности;
* Работа со строками;
* Работа с данными;
* Реализация SQL-запросов;
* Работа с XML;
* Создание стандартных проектов Windows Forms;
* Создание веб-приложений с помощью Web Forms;
* Создание компонентов-сервисов с помощью Web Services;
* И многое другое.

## Обзор языка программирования C#

C# - один из популярных объектно-ориентированных языков программирования. Данный язык также позволяет создавать компоненты.

C# является языком со статической типизацией. Также он полностью отвечает принципам ООП, позволяет явно указывать доступ к элементам, поддерживает наследование классов и интерфейсов, абстракции, перегрузку операторов. Позволяет реализовывать многопоточность.

Основным преимуществом для решения поставленной задачи является поддержка LINQ-запросов на языке C#. Такой тип запросов позволяет с легкостью обращаться к базе данных прямо из кода программы, что сокращает время, затрачиваемое на разработку.

Также C# содержит инструменты, позволяющие управлять версиями зависимых библиотек. Это преимущество позволяет быстрее корректировать программу при выходе новой версии библиотеки или компонента.

Помимо названных преимуществ, язык C# имеет следующие достоинства:

* Быстрое развитие языка, благодаря поддержке компанией Microsoft;
* Типобезопасность языка;
* Синтаксический сахар, позволяющий аккуратно писать код;
* Исчерпывающая справочная информация;
* Возможность разработки как настольных, так и веб-приложений [27].

## Обзор технологии ASP.Net MVC

ASP.NET – это технология, которая позволяет разрабатывать веб-приложения, используя язык C# и платформу Microsoft.NET. В отличие от ASP, ASP.NET является полноценной платформой для создания сложных, но при этом быстрых веб-приложений [28].

Данная технология содержит в себе три платформы для создания веб-приложений: ASP.NET MVC, ASP.NET WebPages, ASP.NET Web API. Рассмотрим подробнее ASP.NET MVC.

Платформа ASP.NET MVC позволяет с легкостью создать масштабное веб-приложение с помощью известного паттерна Model-View-Controller, схема взаимодействия компонентов приведена на Рисунке 3.2. Данный подход имеет свои преимущества, такие как простота повторного использования кода, гибкая разработка, удобное разграничение частей, что позволяет с наименьшими усилиями разработать крупное веб-приложение, а также быстро и качественно обновлять его, добавлять новые функции и компоненты.

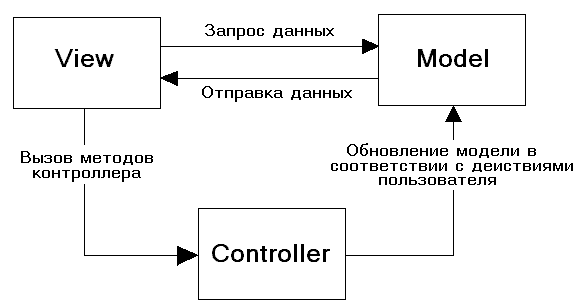


Рисунок 3.2 – Взаимодействие компонентов паттерна MVC

ASP.NET MVC поддерживает TDD-разработку (разработку через тестирование), что значительно сокращает время и увеличивает качество разработки [29].

По сравнению с ASP.NET WebForms, ASP.NET MVC имеет ряд преимуществ, среди которых: мощная маршрутизация URL, удобные для чтения и понимания URL-адреса, автоматически генерирующийся соответствующий стандартам разработки код разметки, автоматическое добавление популярных библиотек, таких как Bootstrap и jQuery.

Помимо названных выше, ASP.NET MVC имеет следующий ряд достоинств:

* Четкое разделение ответственности за счет использования шаблона проектирования MVC;
* Благодаря независимости компонентов, входящих в MVC Framework, их легко заменять на другие компоненты, что позволяет создавать веб-приложения с самым специфичным функционалом;
* Элегантное оформление проекта по умолчанию. Благодаря этой возможности не нужно тратить много времени на приятный дизайн, а можно больше сосредоточиться на функциональности приложения;
* Легкое тестирование приложения;
* Открытый для чтения и изменения код [30].

## Обзор технологии ASP.NET Web API

ASP.NET Web API – это платформа для создания веб-приложений, построенная на тех же принципах, что и ASP.NET MVC, но в отличие от этой технологии, Web API предназначен для создания простых веб-служб с легкой конфигурацией и минимальным количеством кода.

Данная платформа поддерживает принципы HTTP веб-коммуникации и использует основные методы HTTP: GET, POST, PUT, DELETE. Это обусловлено тенденцией повсеместного использования HTTP протокола.

С помощью ASP.NET Web API можно отправлять данные различным клиентам, независимо от них, будь то настольное или веб-приложение [31].

# Выбор метода хранения данных

Серверное приложение информационной системы подразумевает хранение данных. Одним из наилучших решений являются реляционные базы данных. Так как для работы информационной системы нужен большой объем информации в относительно быстрой доступности, используется Entity Framework и VisualStudio SQL Server Data Tools.

## Обзор Visual Studio SQL Server Data Tools

SQL Server Data Tools – это технология, позволяющая создавать реляционные базы данных прямо из среды разработки Visual Studio. Данное средство поддерживает создание следующих баз данных:

* Базы данных SQL Server;
* Модели данных Analysis Services(AS);
* Пакеты Integration Services (IS);
* Отчеты Reporting Services (RS);
* Базы данных SQL Azure.

Основным преимуществом данной технологии является управление базой данных напрямую в Visual Studio с помощью известных средств разработки, таких как IntelliSense, навигация по коду, поддержка языков C# и Visual Basic.

При этом, обозреватель объектов SQL Server представляет объекты базы данных так же, как в SQL Server Management Studio. Достоинством технологии SQL Server Data Tools является поддержка создания, изменения, переименования и удаления таблиц, хранимых процедур, типов и функций привычным методом работы с базой данных, тем самым ускоряет освоение данной технологии. Кроме этого, технология поддерживает сравнение схем, выполнение запросов с помощью контекстного меню в обозревателе SQL Server [32].

## Обзор Entity Framework

Платформа Entity Framework представляет собой набор технологий ADO.NET, обеспечивающих разработку приложений, связанных с обработкой данных. Архитекторам и разработчикам приложений, ориентированных на обработку данных, приходится учитывать необходимость достижения двух совершенно различных целей. Они должны моделировать сущности, связи и логику решаемых бизнес-задач, а также работать с ядрами СУБД, используемыми для сохранения и получения данных. Данные могут распределяться по нескольким системам хранения данных, в каждой из которых применяются свои протоколы, но даже в приложениях, работающих с одной системой хранения данных, необходимо поддерживать баланс между требованиями системы хранения данных и требованиями написания эффективного и удобного для обслуживания кода приложения.

В Entity Framework разработчики получают возможность работать с данными, представленными в форме относящихся к конкретным доменам объектов и свойств, таких как клиенты и их адреса, не будучи вынужденными обращаться к базовым таблицам и столбцам базы данных, где хранятся эти данные. Entity Framework дает разработчикам возможность работать с данными на более высоком уровне абстракции, создавать и сопровождать приложения, ориентированные на работу с данными, одновременно с этим сокращая объем кода, по сравнению с традиционными приложениями. Так как Entity Framework входит в состав .NET Framework, Entity Framework приложения могут выполняться на любом компьютере, на котором установлена платформа .NET Framework, начиная с версии 3.5 с пакетом обновления 1.

Многолетним и общим подходом к разработке является подход, при котором построение приложения или службы представляет собой его разделение на три части: модель домена, логическую модель и физическую модель. Модель домена определяет сущности и связи в моделируемой системе. Логическая модель для реляционной базы данных обеспечивает нормализацию сущностей и связей в целях создания таблиц с ограничениями внешнего ключа. В физической модели учитываются возможности конкретной системы обработки данных путем определения зависящих от ядра базы данных подробных сведений о хранении данных, которые касаются секционирования и индексирование.

Физическая модель совершенствуется администраторами базы данных в целях повышения производительности, но программисты, которые разрабатывают код приложения, в основном вынуждены ограничиваться работой с логической моделью, подготавливая SQL-запросы и вызывая хранимые процедуры. Модели домена в основном используются как инструмент для представления и обмена мнениями о требованиях к приложению, поэтому чаще всего служат в качестве практически не изменяющихся схем, которые рассматриваются и обсуждаются на ранних стадиях проекта, после чего выходят из сферы внимания. Во многих коллективах разработчиков принято пропускать этап создания концептуальной модели и начинать с определения таблиц, столбцов и ключей в реляционной базе данных.

Entity Framework придает значимость моделям, позволяя разработчикам выполнять запросы к сущностям и связям в модели предметной области (вызывается концептуальной модель Entity Framework), опираясь на Entity Framework для их преобразования операции команды, специфического для источника данных. Это позволяет отказаться от применения в приложениях жестко заданных зависимостей от конкретного источника данных [33].

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

Database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных

Model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.

Code first: разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы [34].

При работе в режиме Code First концептуальная модель сопоставлена с режимом хранения в коде. Entity Framework может вывести концептуальную модель, основанную на типах объектов и дополнительных конфигурациях, которые можно задать. Метаданные сопоставления формируются во время выполнения на основе сочетания определений типов домена и дополнительной информации о конфигурации, которая указана в коде. Entity Framework при необходимости создает базу данных на основе метаданных.

При работе со средствами работы с моделью EDM концептуальная модель, модель хранения и сопоставление между ними выражены в схемах на основе XML и определены в файлах с именами с соответствующими расширениями.

Модель хранения и сопоставления при необходимости могут быть изменены без изменения концептуальной модели, классов данных и кода приложения. Модели хранения зависят от поставщика, поэтому можно работать с согласованной концептуальной моделью через различные источники данных.

Entity Framework модели и сопоставления файлов для создания, чтения, обновления и удаления, выполняемых над сущностями и связями концептуальной модели, в эквивалентные операции в источнике данных. Entity Framework поддерживает даже сопоставление сущностей в концептуальной модели с хранимыми процедурами в источнике данных.

При использовании объектно-ориентированного программирования для взаимодействия с системами хранения данных возникают сложности. Безусловно, организация классов часто напоминает организацию таблиц реляционной базы данных, но такое соответствие неидеально. Несколько нормализованных таблиц часто соответствуют единственному классу, а связи между классами представлены не так, как связи между таблицами [33].

# Структура серверной части информационной системы

Серверная часть информационной системы обеспечивает взаимодействие базы данных и клиентских приложений, подготовка данных к представлению, бизнес-логику. В основе серверной части лежит трехслойная архитектура, структура которой приведена на Рисунке 3.3, содержащая следующие слои:

* Data Access Layer (DAL), отвечающая за создание и манипуляции с базой данных. На данном этапе используется Entity Framework с подходом Code First;
* Business Logic (BL), отвечающая за взаимодействие базы данных и представлений, а также для реализации дополнительных функций, таких как журналирование;
* Presentation Layer (PL), в качестве которого выступает веб-сервис, основанный на ASP.NET Web API, является слоем, позволяющим формировать представления для обращающихся клиентских приложений. На данном этапе используется реализация REST, позволяющая обращаться к серверу с помощью стандартных HTTP методов. Данный подход позволяет не зависеть от типов клиентских приложений.

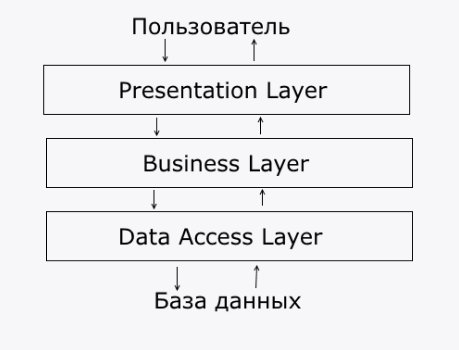


Рисунок 3.3 – Структура трехслойной архитектуры

Помимо Web API сервиса в серверную часть информационной системы входит клиентское веб-приложение для администратора системы, позволяющее редактировать справочники, проверять записи для базы технологических возможностей, составлять отчеты по базе технологических возможностей и деталям.

Структура проекта серверной части информационной системы представлена на Рисунке 3.4:

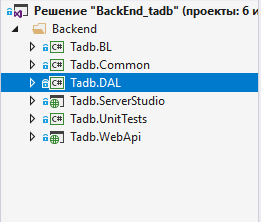


Рисунок 3.4 – Структура проекта

Основой серверной части информационной системы учета технологических возможностей механообрабатывающего предприятия является база данных. Всю базу данных можно поделить на 4 части: административная часть, часть, отвечающая за базу технологических возможностей, часть, отвечающая за хранение информации о детали, часть, хранящая маршрутные карты.

Административная часть включает в себя следующие сущности: Company (Компания), Employee (Сотрудник), Role (Роль сотрудника). Она нужна для того, чтобы хранить данные о сотрудниках, которые вносят записи, разграничить доступ по ролям пользователей, разграничить доступ по организациям и филиалам организаций.

Для Сотрудников разработаны 4 роли: Администратор, Технолог по базе технологических возможностей, Технолог по составлению маршрутных карт, Удаленная запись.

Администратор имеет доступ ко всему функционалу информационной системы, а также к веб-приложению, с помощью которого может изменять статус записи (принимать или отклонять новую запись), редактировать справочники, просматривать журнал и объемы доступных записей в базе данных.

Часть, отвечающая за базу технологических возможностей предприятия, позволяет хранить данные о всем оборудовании конкретного предприятия, а также о том, какие операции можно произвести на данном оборудовании. Данная часть включает в себя следующие сущности: Record (Запись о технологической возможности), Status (Статус записи), а также четыре справочника: ProcessCatalog (Справочник технологических процессов), EquipmentCatalog (Справочник оборудования), SurfaceCatalog (Справочник поверхностей), FixtureCatalog (Справочник оснастки).

Статус записи позволяет отобразить, можно ли данную запись использовать в документации, а также учитывать их при составлении маршрутных карт.

Статус записи имеет право изменять только администратор системы. Возможны четыре типа статуса записи:

1. Запись на проверку – данный статус устанавливается при добавлении новой записи в базу данных;
2. Проверенная запись – данный статус устанавливается администратором системы после проверки правильности введенных данных;
3. Отклоненная запись – данный статус устанавливается администратором при условии, что запись не прошла проверку. Тогда она отправляется на переработку технологу по базе технологических возможностей;
4. Устаревшая запись – данный статус может присвоить администратор системы при условии, что после проверки оборудования предприятия выявлены некие изменения (ухудшение показателей от времени, улучшение показателей после ремонта, улучшение оборудования, замена некоторых составляющих и так далее).

Часть, отвечающая за хранение информации о детали, впоследствии используется для построения маршрутных карт. Ее также можно разделить на три подраздела: Данные о детали, данные о заготовке, данные об оборудовании для изготовления детали. Следует уточнить, что данные об оборудовании в части хранения информации о детали лишь описывает требования к оборудованию при изготовлении конкретной детали.

Диаграмма базы данных представлена на рисунке 3.5:

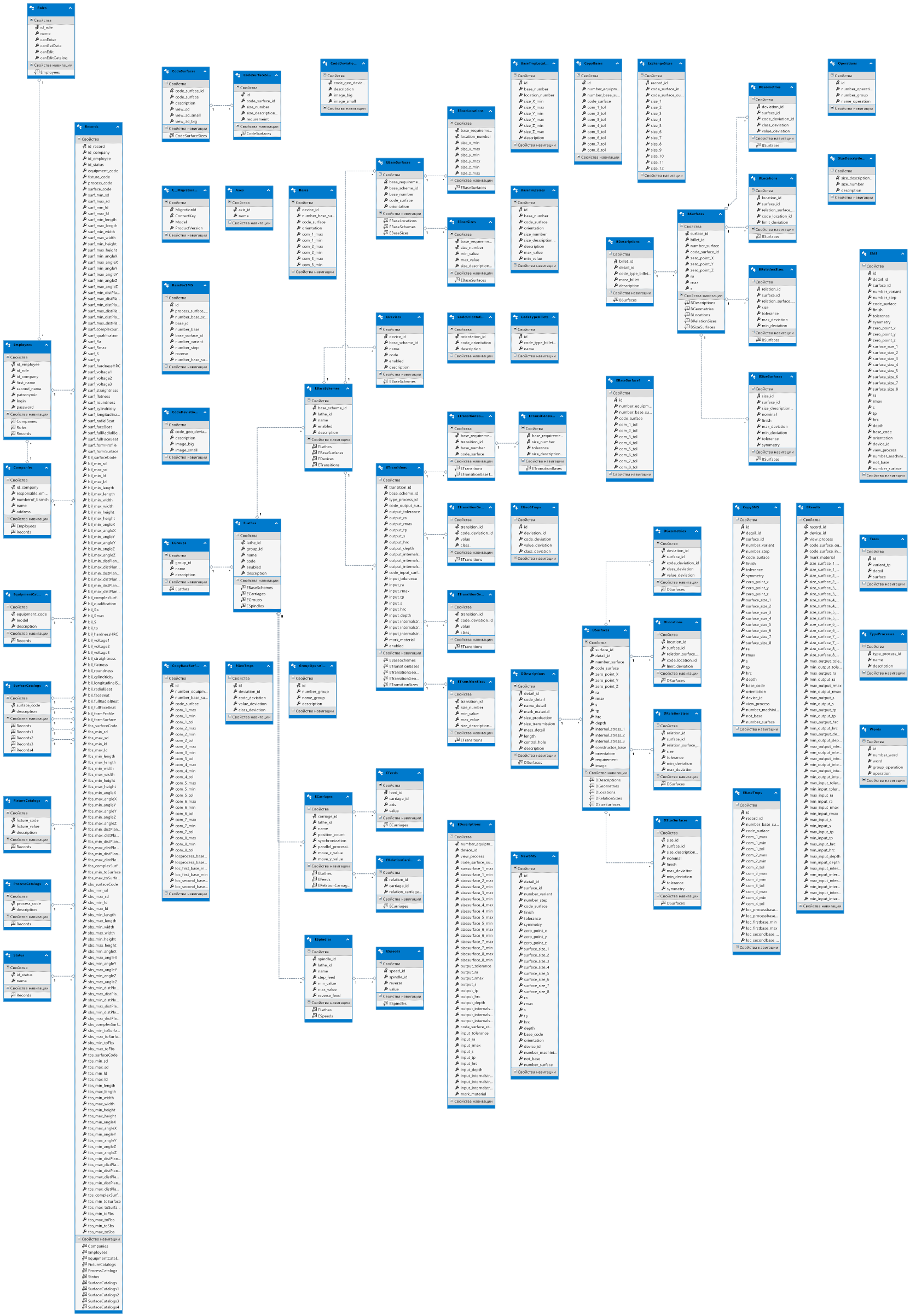
**

Рисунок 3.5 – Диаграмма базы данных

Подраздел данных о детали содержит следующие сущности: DDescription (Описание детали), DGeometry (Требования к отклонениям от геометрической формы), DLocation (Требования к взаимному расположению), DRelationSize (Размерные связи), DSizeSurface (Размерные характеристики), DSurface (Описание базовой поверхности детали).

Подраздел данных о заготовке содержит следующие сущности: BDescription (Описание заготовки), BSurface (Описание базовой поверхности заготовки).

Подраздел данных об оборудовании для изготовления содержит следующие сущности: EGroup (Описание группы оборудования), ELathe (Описание станка), EBaseScheme (Описание схемы базирования).

Помимо этого, данная часть содержит два справочника: CodeGeometry (Справочник отклонений от геометрической формы), CodeLocation (Справочник отклонений от взаимного расположения).

Четвертая часть отвечает за хранение данных технологических карт. Она содержит следующие сущности: Operation (Операция), Route (Маршрут), RouteCard (Маршрутная карта).

Раздел бизнес-логики (Tadb.BL) содержит в себе классы Autorization Service, Import Service и Record Logic.

Класс Autorization Service содержит в себе методы HashPassword(), CreateNewUser(), GetEmployeeByLoginPassword().

Рассмотрим метод HashPassword(). Данный метод принимает строку, которая является паролем пользователя. С помощью String Builder каждый символ хешируется с помощью алгоритма хеширования SHA512. Данный метод возвращает зашифрованную строку.

Метод GetEmployeeByLoginPassword( ) используется для авторизации пользователей в системе. Данный метод принимает на вход строку логина и строку пароля пользователя, после чего вызывает метод HashPassword(), и ищет совпадения записи по логину и паролю в сущности Employee. После чего из сущности передается id роли сотрудника, тем самым определяя его полномочия. Метод возвращает переменную, которая отвечает за существование в базе данного пользователя и передает данные о пользователе клиентскому приложению.

Данный метод имеет обработку исключений на случай, если пользователь ввел неправильный логин или пароль.

Метод CreateNewUser(), позволяет добавлять администратору новых пользователей с хранением в базе данных хешированного пароля. На вход подаются данные о роли и компании, строки: имя, фамилия, отчетство, логин, пароль.

Для строки “password” вызывается метод HashPassword(), происходит проверка на наличия пользователя с таким логином в базе данных, и если такого пользователя нет, формируется запись сущности Employee с хешированной строкой пароля и записывается в базу данных с помощью метода Add().

Класс ImportService содержит методы ParseRecordString() и FillDataBase(), позволяющие быстро внести в базу данных импортированный в клиентской части информационной системы файл \*.csv.

Метод ParseRecordString() позволяет выделять из потока данных нужные и формирует из них пригодную для внесения в базу данных запись. На вход данный метод принимает строковый массив, который был передан от клиентской части при импорте файла. После этого массив разбирается на отдельные строки, из которых формируется запись для сущности Record. Выходными данными метода является экземпляр сущности Record.

Метод FillDataBase() позволяет внести сразу несколько экземпляров сущности Record в базу данных, на случай, если в файле оказалось больше одной строки. На вход также подается строковый массив, полученный от клиентской части информационной системы. Далее массив разбирается на отдельные записи с помощью нахождения знака следующей строки, и каждая строка преобразуется в экземпляр сущности Record. Из полученных экземпляров формируется список сущностей, который впоследствии вносится в базу данных со статусом «на проверку».

При этом в методе FillDataBase() вызывается метод ImportNaming() у экземпляра класса LogHelper, который записывает акт импорта данных в журнал.

Класс RecordLogic содержит методы, позволяющие делать различные выборки по записям в таблице Record. Данный класс содержит три метода: GetRecordsByCompany(), GetRecordsByEmployee(), GetRecordsByStatus().

GetRecordsByCompany() позволяет выбирать все записи, которые относятся определенному предприятию. Данное условие нужно соблюдать для предотвращения утечки информации от одного предприятия к другому. На вход подается ид компании, на выходе формируется список записей в таблице, чей ид совпал с введенным ид.

GetRecordsByEmployee () позволяет выбирать все записи, которые относятся к определенному сотруднику. Чтобы сотрудник мог работать только со своими записями и не изменять записи других сотрудников, следует отображать только записи, автором которых является конкретный авторизованный сотрудник. На вход подается ид сотрудника, на выходе также формируется список записей в таблице, которые принадлежат определенном ид сотрудника.

Метод GetRecordsByStatus() требуется для разграничения ответственности за записи, хранящиеся в БД. Технолог по базе технологических возможностей имеет доступ только к записям, принадлежащим ему как автору, и только со статусами «на проверку» или «отклонено». Для администратора системы будет намного удобнее найти записи с нужным статусом для проверки данных и изменения статуса.

Раздел Common содержит в себе классы, относящиеся к системе журналирования. Данная система реализована с помощью сторонней библиотеки NLog. Использование данной библиотеки оправдано, так как она имеет полезные для разработки данной информационной системы функции, такие как:

* Возможность выбора метода хранения данных – в файле или в базе данных;
* Встроенная передача данных на сервер, что облегчает настройку журналирования для клиентских приложений;
* Удобные встроенные описания событий, которые полностью удовлетворяют потребности в описании событий журнала;
* Настраиваемая приоритетность записей;
* Асинхронная работа логирования;
* Удобная настройка формата записей.

Раздел Common включает в себя два класса: ApplicationData и LogHelper.

Класс ApplicationData является вспомогательным и описывает структуру данных, которая понадобится для добавления записи в файл. Данный класс содержит свойства Login и Role, чтобы с помощью них получать информацию о пользователе, работающем в системе, и на основании них формировать строку для записи действия в журнал.

Класс LogHelper содержит методы, позволяющие добавлять записи в журнал с различными параметрами под каждое событие. Данный класс имеет методы Naming(), ImportNaming(), LogString().

Метод LogString() формирует строку для записи в журнал в соответствии с определенным форматированием. Первой частью идет название события, второй – сотрудник, который запустил это событие, третьей – строка данных, сформированная в других методах (Naming() и ImportNaming()). Данная функция вынесена в отдельный метод, так как она работает одинаково для различных методов. Результатом работы метода является записывание данных в файл журнала.

Метод Naming() позволяет добавлять различные события при различных действиях пользователя. При совершении пользователем действия метод формирует строку с данными о пользователе и времени совершения события. Результатом работы данного метода является строка, которая является подстрокой готовой записи для занесения в журнал.

Метод ImportNaming() отличается от Naming() лишь тем, что он принимает в качестве аргумента строку с названием импортированного файла.

Раздел DAL (Data Access Layer) отвечает за создание и взаимодействие с базой данных. Он содержит классы, описывающие сущности базы данных. С помощью Entity Framework и подхода Code First данные классы создают таблицы и поля.

Класс MachineDbContext описывает взаимодействие сущностей друг с другом и с другими частями кода. Переопределенный метод OnModelCreating() позволяет запретить каскадное удаление, где это необходимо, описать явно связи между сущностями и тому подобное.

Проект WebApi является сервисом, к которому обращаются клиенты. Он основывается на платформе ASP.NET Web API, которая была разработана на базе ASP.NET MVC, поэтому имеет схожую архитектуру. Контроллеры данного проекта представляют собой классы с методами, основанными на структуре REST. Структура проекта представлена на Рисунке 3.6:

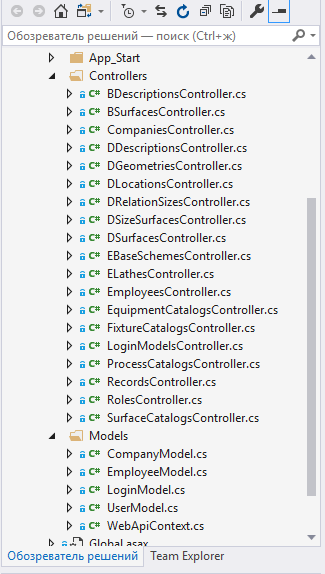


Рисунок 3.6 – Структура проекта WebApi

Каждый контроллер имеет следующие методы:

* GET() – данный метод позволяет возвращать данные пользователю. По умолчанию возвращает все данные, хранящиеся в модели, при указании в адресе определенного ид возвращает только то представление, которое запрашивается;
* POST() – данный метод создает новый ресурс, отправляя данные, описывающие этот ресурс;
* PUT() – данный метод создает или заменяет ресурс;
* DELETE() – данный метод удаляет ресурс по указанному адресу.

Помимо контроллеров, проект содержит некоторые специфичные модели, которые помогают правильно формировать данные, отправляемые клиентскому приложению. Таковыми являются классы CompanyModel, EmployeeModel, LoginModel, UserModel. Класс WebApiContext предоставляет контроллерам доступ к моделям.

Проект ServerStudio является веб-приложением разработанным на базе ASP.NET MVC. Данный проект реализует архитектуру Model-View-Controller.

Проект содержит в себе следующие страницы:

* Главная страница;
* Отчет по базе технологических возможностей;
* Отчет по деталям;
* Справочник оборудования;
* Справочник приспособлений;
* Справочник технологических переходов;
* Справочник поверхностей.

Данный проект позволяет администратору системы добавлять необходимые справочные материалы, проверять записи в базе технологических возможностей, контролировать работу сотрудников и проверять журналы системы.

Вид домашней страницы представлен на Рисунке 3.7:

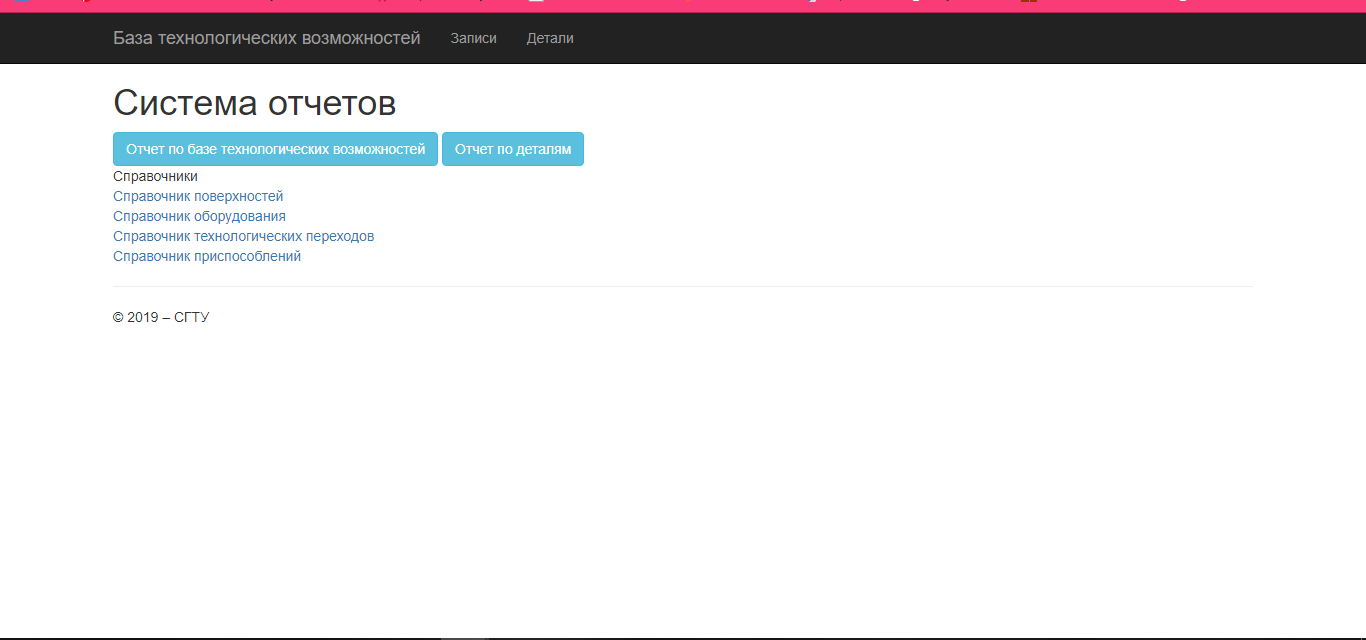


Рисунок 3.7 – Домашняя страница веб-приложения

Данная страница содержит ссылки на весь функционал приложения для того, чтобы администратор системы быстро находил нужный раздел и отчет.

Отчет по базе технологических возможностей представляет собой таблицу со следующими полями:

* Сотрудник – автор записи;
* Статус записи;
* Код оборудования;
* Код приспособления;
* Код технологического перехода;
* Код обрабатываемой поверхности.

Данный отчет служит для контроля заполненности баз учета технологических возможностей, а также позволяет отследить работу сотрудников.

Вид страницы с отчетом по базе технологических возможностей представлен на Рисунке 3.8:

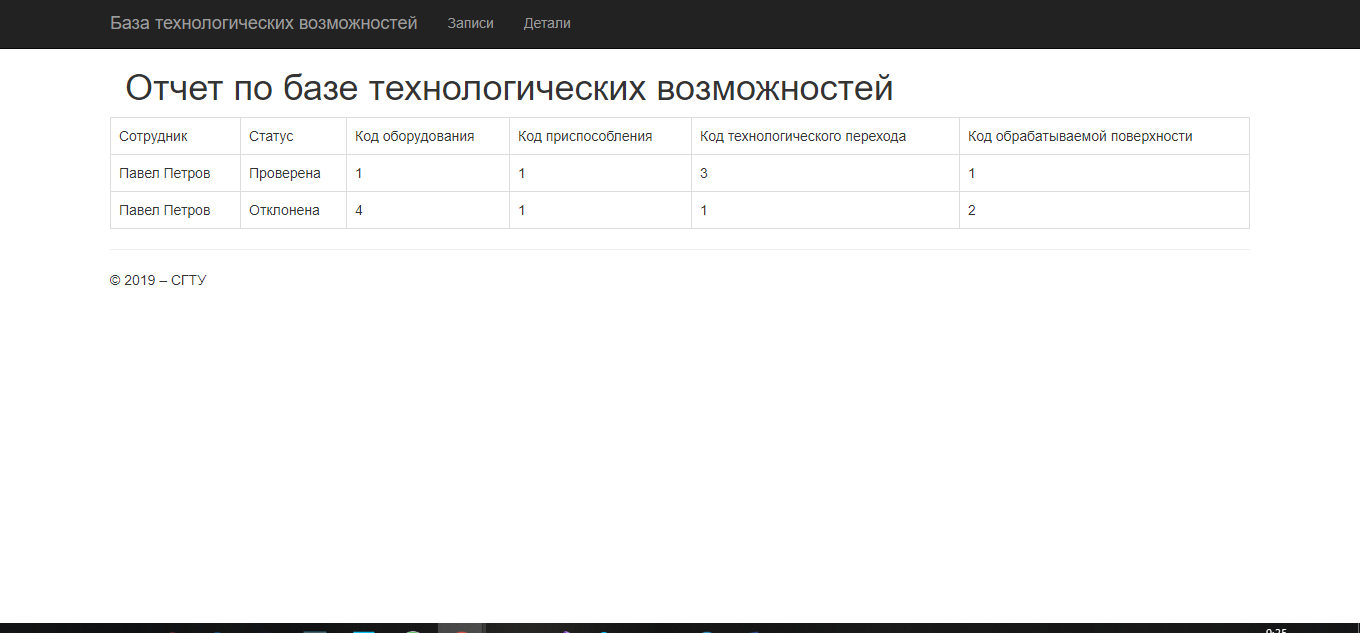


Рисунок 3.8 – Страница отчета по базе технологических возможностей

Отчет по базе деталей представляет себе таблицу с полным перечнем деталей, содержащих общую информацию. Данный отчет может использоваться для контроля заполненности базы данных, а также для учета объемов партий изготовления.

Страница с отчетом по базе деталей представлена на Рисунке 3.9:

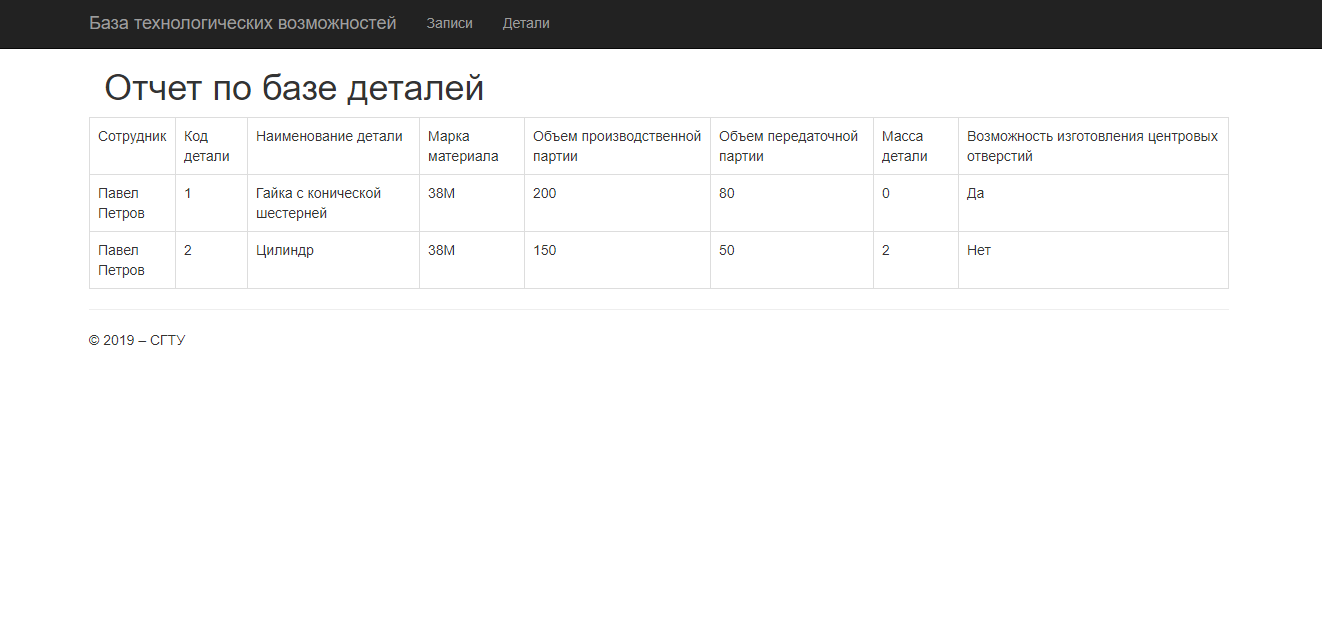


Рисунок 3.9 – Страница отчета по базе деталей

Также веб-приложение содержит в себе информацию по справочникам. Справочник обрабатываемых поверхностей представлен на Рисунке 3.10:

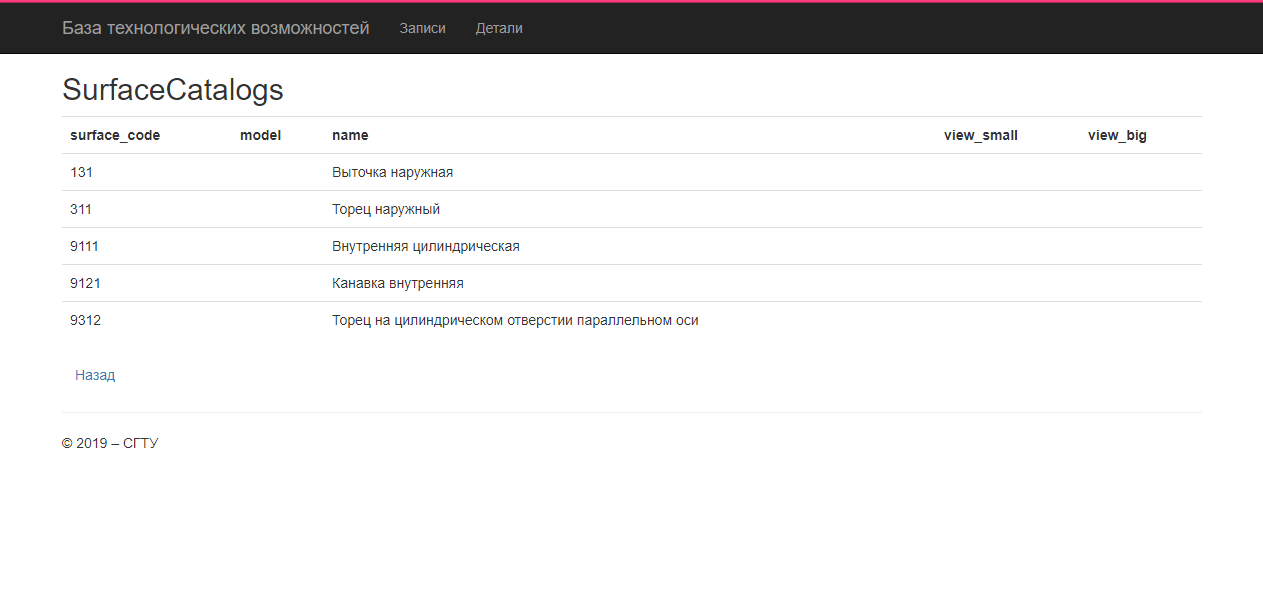


Рисунок 3.10 – Страница справочника обрабатываемых поверхностей

Справочник оборудования представлен на Рисунке 3.11:

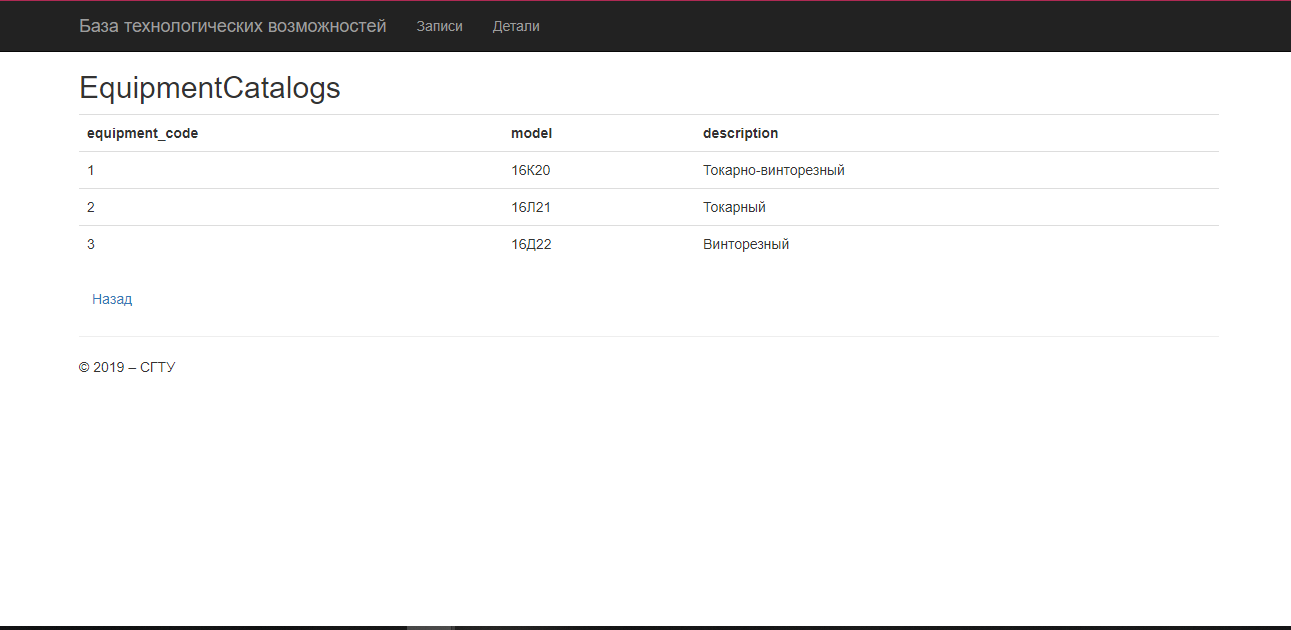


Рисунок 3.11 – Страница справочника оборудования

Справочник технологических переходов представлен на Рисунке 3.12:

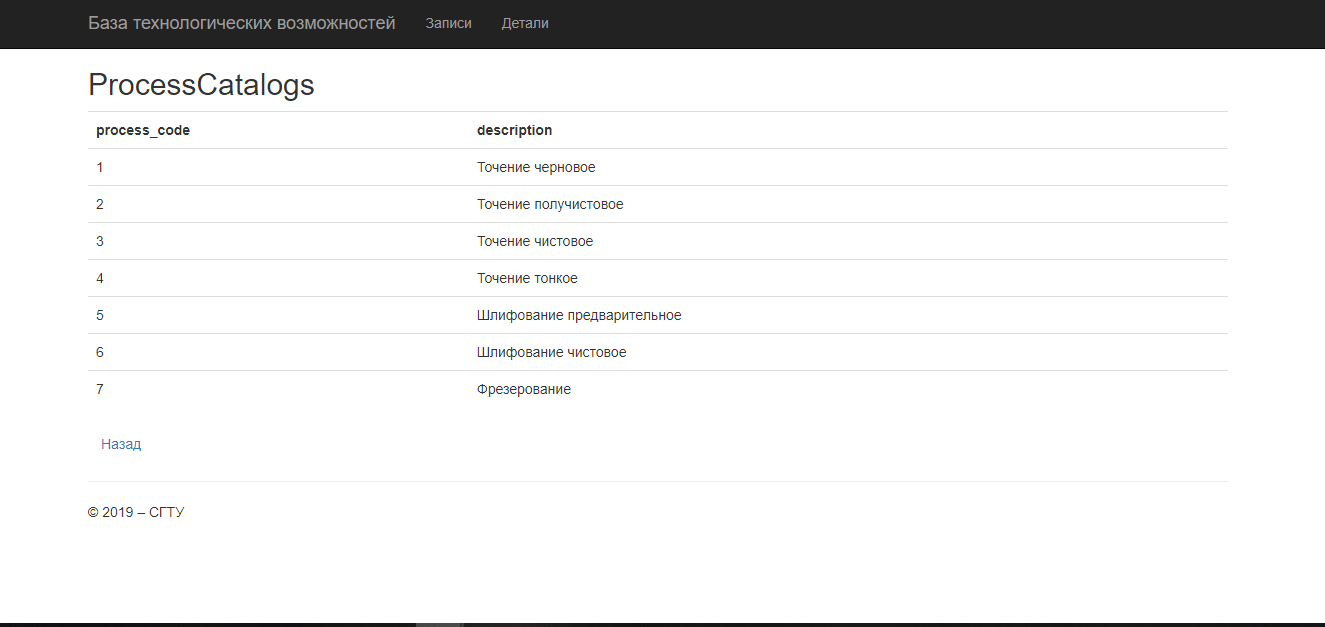


Рисунок 3.12 – Страница справочника технологических переходов

Справочник приспособлений представлен на Рисунке 3.13:

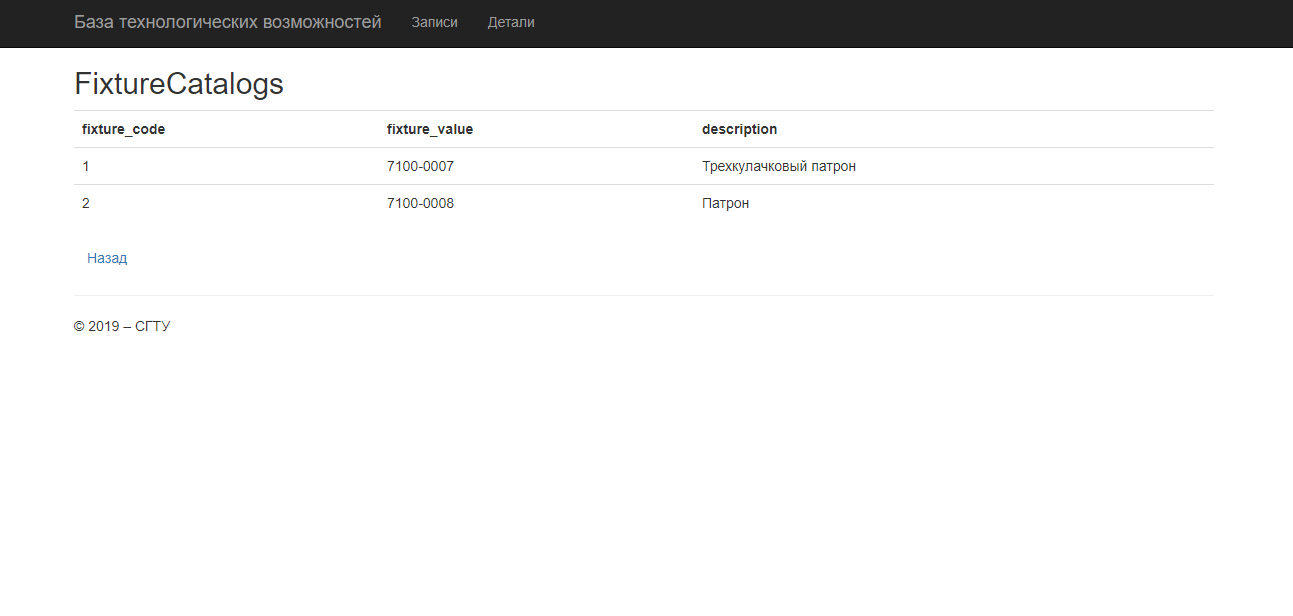


Рисунок 3.13 – Страница справочника приспособлений

# Заключение

При выполнении выпускной квалификационной работы была разработана серверная часть информационной системы учета технологических возможностей механообрабатывающего оборудования. Разработанная система отвечает поставленным требованиям к информационной системе.

Серверная часть информационной системы разработана на языке программирования C# для платформы Microsoft.NET. Во время разработки использовалась база данных на SQL Server Express. Для разработки веб-приложений использовались технологии ASP.NET MVC 5 и ASP.NET Web API.

# Список литературы

1. Картамышева Е. С., Иванченко Д. С. Промышленная автоматизация в России: проблемы и их решения // Молодой ученый. — 2016. — №28. — С. 93-95.
2. Шестаков Н. В., Мишин С. П. Повышение эффективности промышленных предприятий России за счёт передовых решений в автоматизации // Автоматизация в промышленности. — № 3, 2016. – С. 3-5.
3. Автоматизация проектирования технологических процессов: учеб. пособие для вузов [электронный ресурс] / В.И. Аверченков, Ю.М. Казаков. – 2-е изд., стереотип. – М.: ФЛИНТА, 2011. – 229 с.
4. 4. Волков И.А. Кузьмина Т.А. Масягин В.Б. Примак Д.Д. Автоматизация проектирования технологического процесса механической обработки с применением математического моделирования // Естественные и технические науки. - 2014.
5. Обзор системы «1С: Предприятие 8» [Электронный ресурс]. – 2019, URL: http://v8.1c.ru/overview/ (Дата обращения: 24.03.2019)
6. SAP ERP Review [Электронный ресурс]. – 2019, URL: <https://reviews.financesonline.com/p/sap-erp/> (Дата обращения: 24.03.2019)
7. Что такое SAP системы [Электронный ресурс]. – 2019, URL: <http://asapcg.com/press-center/articles/chto-takoe-sap-sistemy/> (Дата обращения: 22.03.2019)
8. Комплексная система автоматизации технологической подготовки производства TECHCARD 4.1 [Электронный ресурс]. – 2019, URL: <https://sapr.ru/article/8228> (Дата обращения: 24.03.2019)
9. Techcard [Электронный ресурс]. – 2019, URL: <http://www.intermech.ru/techcard.htm> (Дата обращения: 14.04.2019)
10. CAD/CAM системы [Электронный ресурс]. – 2019, URL: <https://www.autodesk.ru/solutions/cad-cam> (Дата обращения: 29.04.2019)
11. CAPP [Электронный ресурс]. – 2019, URL: <http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:CAPP> (Дата обращения: 29.04.2019)
12. Продукты [Электронный ресурс]. – 2019, URL: <https://adem.ru/products/> (Дата обращения: 14.04.2019)
13. Cimatron [Электронный ресурс]. – 2019, URL: <https://ru.3dsystems.com/software/cimatron> (Дата обращения: 16.04.2019)
14. RTM Changes the Way It Produces Parts with Cimatron CAD/CAM software [Электронный ресурс]. – 2019, URL: <https://ru.3dsystems.com/customer-stories/rtm-changes-way-it-produces-parts-cimatron-cadcam-software> (Дата обращения: 16.04.2019)
15. Описание CAD/CAM системы CimatronE [Электронный ресурс]. – 2019, URL: <http://planetacam.ru/choice/cimatrone/> (Дата обращения: 16.04.2019)
16. Цуканова О. А. Методология и инструментарий моделирования бизнеспроцессов: учебное пособие – СПб.: Университет ИТМО, 2015. – С. 15-21.
17. Репин В. В., Елиферов В. Г. Процессный подход к управлению. Моделирование бизнес-процессов – М.: Манн, Иванов и Фербер, 2013 – С. 99-109.
18. Цуканова О. А. Методология и инструментарий моделирования бизнеспроцессов: учебное пособие – СПб.: Университет ИТМО, 2015. – С. 57- 61.
19. Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд. Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – С. 28-47.
20. Лекция 3. Виды диаграмм UML [Электронный ресурс]. – 2019, URL: <https://www.intuit.ru/studies/courses/1007/229/lecture/5954?page=2> (Дата обращения: 17.04.2019)
21. Диаграммы прецедентов и их нотация. [Электронный ресурс]. – 2019, URL: <https://www.intuit.ru/studies/courses/1007/229/lecture/5962?page=2> (Дата обращения: 17.04.2019)
22. Диаграммы взаимодействия (последовательности и коммуникации) [Электронный ресурс]. – 2019, URL: <https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema13/tema13_3> (Дата обращения: 17.04.2019)
23. Преимущества Microsoft.Net Framework [Электронный ресурс]. – 2019, URL: <https://support.microsoft.com/ru-ru/help/929300/benefits-of-the-microsoft-net-framework> (Дата обращения: 04.05.2019)
24. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. — СПб.: Питер, 2013. — 896 с.: ил. — (Серия «Мастер-класс»).
25. Пономарев В. Программирование на C++/C# в Visual Studio .NET 2003. – СПб.: БХВ-Петербург, 2004. – C. 2-4.
26. Основные компоненты .NET Framework [Электронный ресурс]. – 2019, URL: <https://codeby.net/threads/osnovnye-komponenty-net-framework-clr-i-framework-class-library.65650/> (Дата обращения: 04.06.2019)
27. 1.Платформа .NET. Основные возможности и достоинства. CLR. MSIL. Особенности .NET Framework 2.0. Microsoft Visual Studio 2005. [Электронный ресурс]. – 2019, URL: <http://www.konspektov.net/question/4056> (Дата обращения: 04.05.2019)
28. Краткий обзор языка C#. [Электронный ресурс]. – 2019, URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/> (Дата обращения: 07.05.2019)
29. Особенности и преимущества языка C#. [Электронный ресурс]. – 2019, URL: <https://studbooks.net/2070063/informatika/osobennosti_preimuschestva_yazyka> (Дата обращения: 07.05.2019)
30. ASP.NET [Электронный ресурс]. – 2019, URL: <https://professorweb.ru/my/ASP_NET/base/level1/aspnet_info.php> (Дата обращения: 07.05.2019)
31. Обзор ASP.NET [Электронный ресурс]. – 2019, URL: <https://docs.microsoft.com/ru-ru/aspnet/overview> (Дата обращения: 07.05.2019)
32. Преимущества ASP.NET MVC [Электронный ресурс]. – 2019, URL: <https://professorweb.ru/my/ASP_NET/mvc/level1/1_2.php> (Дата обращения: 07.05.2019)
33. Что такое WebAPI? [Электронный ресурс]. – 2019, URL: <https://www.smarly.net/asp-net-mvc-4-in-action/mastering-asp-net-mvc/asp-net-web-api/what-is-web-api> (Дата обращения: 07.05.2019)
34. SQL Server Data Tools (SSDT) [Электронный ресурс]. – 2019, URL: <https://docs.microsoft.com/ru-ru/sql/ssdt/sql-server-data-tools?view=sql-server-2017> (Дата обращения: 07.05.2019)
35. Общие сведения об Entity Framework [Электронный ресурс]. – 2019, URL: <https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/ef/overview> (Дата обращения: 07.05.2019)
36. Что такое Entity Framework [Электронный ресурс]. – 2019, URL: <https://metanit.com/sharp/entityframework/1.1.php> (Дата обращения: 07.05.2019)