МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

РАЗРАБОТКА МНОГОПОЛЬЗОВАТЕЛЬСКОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ОПТИМИЗАЦИИ И УСКОРЕНИЯ ПРОЦЕССА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Курсовое проектирование

по дисциплине «Базы и банки данных»

КР.1-53 01 02.10028412.15.81-01

Исполнитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Самусев Данила Андреевич рррррррррррррррррррррррррррррррррррррррррррррррррррррррАСОИ-181

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мрочек Татьяна Владимировна

(подпись)

Дата допуска к защите \_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты \_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_

Могилев 2021

Содержание

[Введение 4](#_Toc71565530)

[1 Техническое задание 5](#_Toc71565531)

[1.1 Анализ предметной области (данные по оргштатной структуре предприятия, информация о принятых технологиях деятельности) 5](#_Toc71565532)

[1.2 Постановка задачи (стратегические направления и перспективы развития предприятия) 5](#_Toc71565533)

[2 Технический проект информационной системы 8](#_Toc71565534)

[2.1 Разработка структурной диаграммы программного модуля и её описание 8](#_Toc71565535)

[2.1.1 Контекстная диаграмма и диаграммы детализации процессов 8](#_Toc71565536)

[2.1.2 Диаграмма дерева узлов 14](#_Toc71565537)

[2.2 Информационная модель 15](#_Toc71565538)

[2.2.1 Идентификация сущностей и связей. ER-диаграмма логического уровня 15](#_Toc71565539)

[2.2.2 ER-диаграмма физического уровня. Ограничения ссылочной целостности. Переопределение триггеров. Индексирование отношений. 18](#_Toc71565540)

[2.2.3 Определение представлений, хранимых процедур серверной компоненты 19](#_Toc71565541)

[2.3 Верификация спроектированной логической модели 20](#_Toc71565542)

[3 Реализация системы 22](#_Toc71565543)

[3.1 SQL-определения регламентированных запросов 22](#_Toc71565544)

[3.2 SQL-определения триггеров 24](#_Toc71565545)

[3.3 SQL-определения хранимых процедур 26](#_Toc71565546)

[3.4 SQL-определения курсоров 29](#_Toc71565547)

[3.5 Распределение прав доступа пользователей 30](#_Toc71565548)

[4 Результаты тестирования информационной системы 33](#_Toc71565549)

[Заключение 42](#_Toc71565550)

[Список использованных источников 43](#_Toc71565551)

Введение

Целью данной работы является проектирование многопользовательской информационной системы. Систему предполагается использовать на рабочих местах членов команды разработки для повышения производительности и скорости разработки. Приложение имеет несколько преимуществ по сравнению со своими аналогами: простота в освоении, гибкость, масштабируемость. Данное приложение может быть использовано небольшими командами разработчиков или, например, организаторами практически любого мероприятия. Пояснительная записка к курсовому проектированию содержит 4 раздела:

* Техническое задание – раздел содержит всю необходимую информацию по предметной области, а тек же постановку задачи.
* Технический проект информационной системы – раздел содержит функциональную и информационную модель информационной системы.
* Реализация системы – содержит SQL-определения, применимые в рамках поставленной задачи.
* Результаты тестирования информационной системы – содержит описание различных тестов, которые проходила информационная система.
* Заключение – содержит обобщение выполненной работы.

1 Техническое задание

1.1 Анализ предметной области (данные по оргштатной структуре предприятия, информация о принятых технологиях деятельности)

В качестве объекта автоматизации выбран процесс разработки программного обеспечения.

Система создаётся на основе следующих документов:

* список задач для реализации;
* список ошибок для исправления;

список реализованных задач для тестирования;

1.2 Постановка задачи (стратегические направления и перспективы развития предприятия)

Проектирование многопользовательской информационной системы. Систему предполагается использовать на рабочих местах членов команды разработки для повышения производительности и скорости разработки. Для управления тестированием могут быть интегрированы различные расширения, позволяющие совместить процессы тестирования и разработки программного обеспечения. Команды могут разбить тестирование и разработку на этапы и повысить его эффективность. Команды контроля качества используют задачи, настраиваемые экраны, поля и процессы для управления ручным и автоматическим тестированием.

Систему предполагается создать с целью упростить управление рабочим процессом для самых различных команд, для улучшения качества обслуживания заказчика.

Критерий оценки достижений целей системы – увеличение числа выполненных задач. Т.к. система позволит увеличить скорость разработки, то увеличится количество реализованного функционала.

Система должна удовлетворять следующим требованиям:

* Надежности;
* Безопасности;
* Требования к защите информации от несанкционированного доступа. Должна быть предусмотрена защита от несанкционированного доступа к данным, ввода данных, их удаления;
* Система должна быть доступна с любого компьютера в сети компании или команды разработки;
* Информация, хранящаяся в системе, должна быть защищена от аварийных ситуаций, влияния внешних воздействий (радиоэлектронная защита);
* Информация, хранящаяся в системе, должна быть защищена от аварийных ситуаций, влияния внешних воздействий (радиоэлектронная защита);
* Квалификация персонала. Персонал должен быть обучен правилам работы с системой, при этом не обязательно наличие специального технического образования;

Функции, выполняемые подсистемами объектов автоматизации:

* + Отображение задач в различных статусах. Без использования автоматизированной системы поиск активных приоритетных задач занимает значительное время. И для хранения самих формуляров необходимы дополнительные площади. Автоматизированная система хранит эту информацию сколь угодно долго, поиск осуществляется за считанные секунды, не требуется дополнительная площадь.
  + Создание новых задач, контроль их статуса. При использовании автоматизированной системы контроль над текущими задачами будет осуществляться легко и быстро.
  + Проверка задач, закрепленных за работником. Автоматизированная система хранит информацию о задачах сколь угодно долго, поиск и проверка будет осуществляться за считанные секунды. Работник также сможет видеть список задач, закрепленных за ним.
  + Перевод задач в различные статусы позволяет каждому члену команды разработки концентрироваться только на ему нужных задачах.

Программное обеспечение системы не должно зависеть от аппаратных средств компьютера. Необходимое программное обеспечение:

* MS SQL Server 2019, .Net Framework, Windows 7/10.

2 Технический проект информационной системы

2.1 Разработка структурной диаграммы программного модуля и её описание

2.1.1 Контекстная диаграмма и диаграммы детализации процессов

Контекстная диаграмма представлена на Рисунке 1.

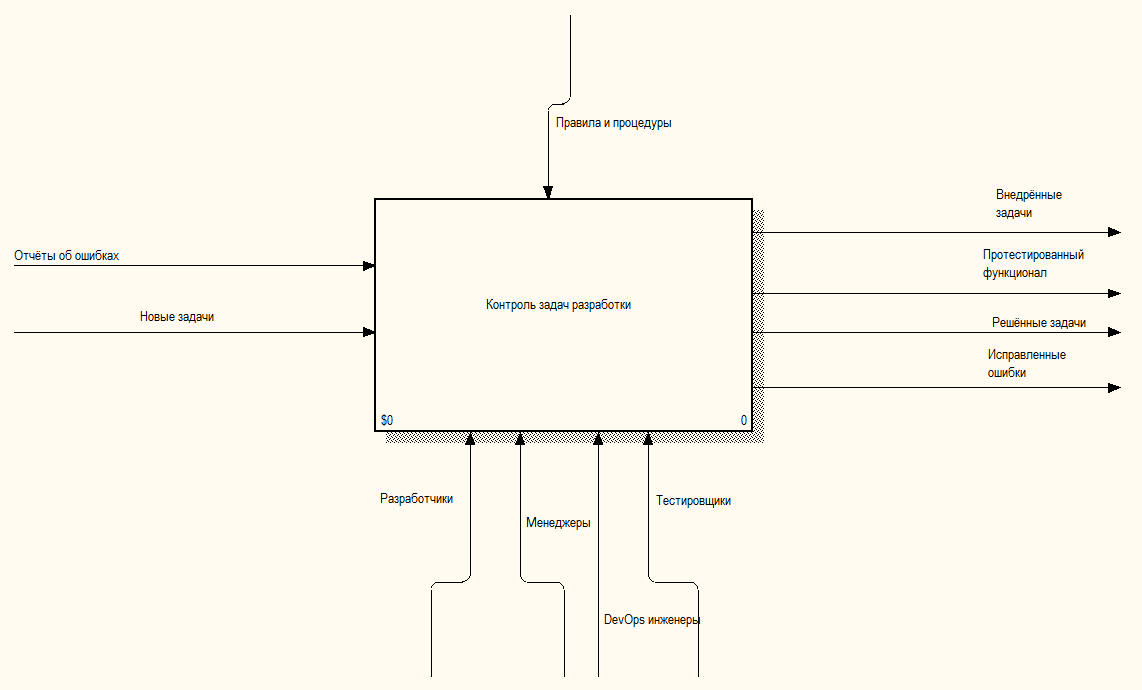


Рисунок 1 – Контекстная диаграмма.

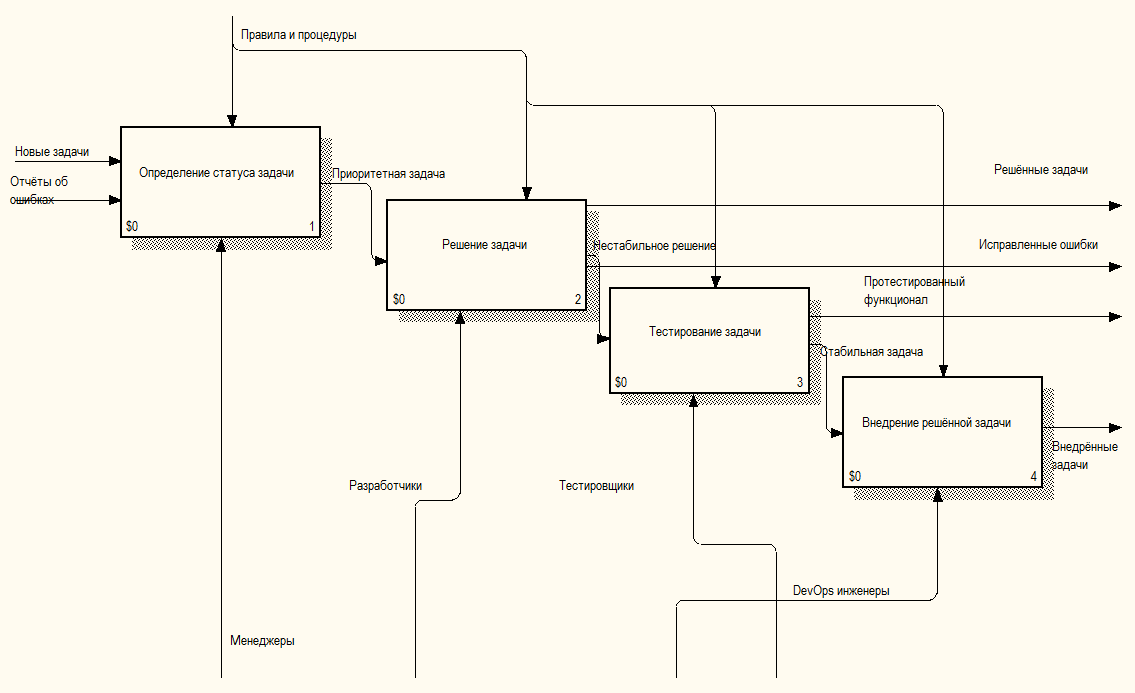


Рисунок 2 – Декомпозиция процесса «Контроль задач разработки»

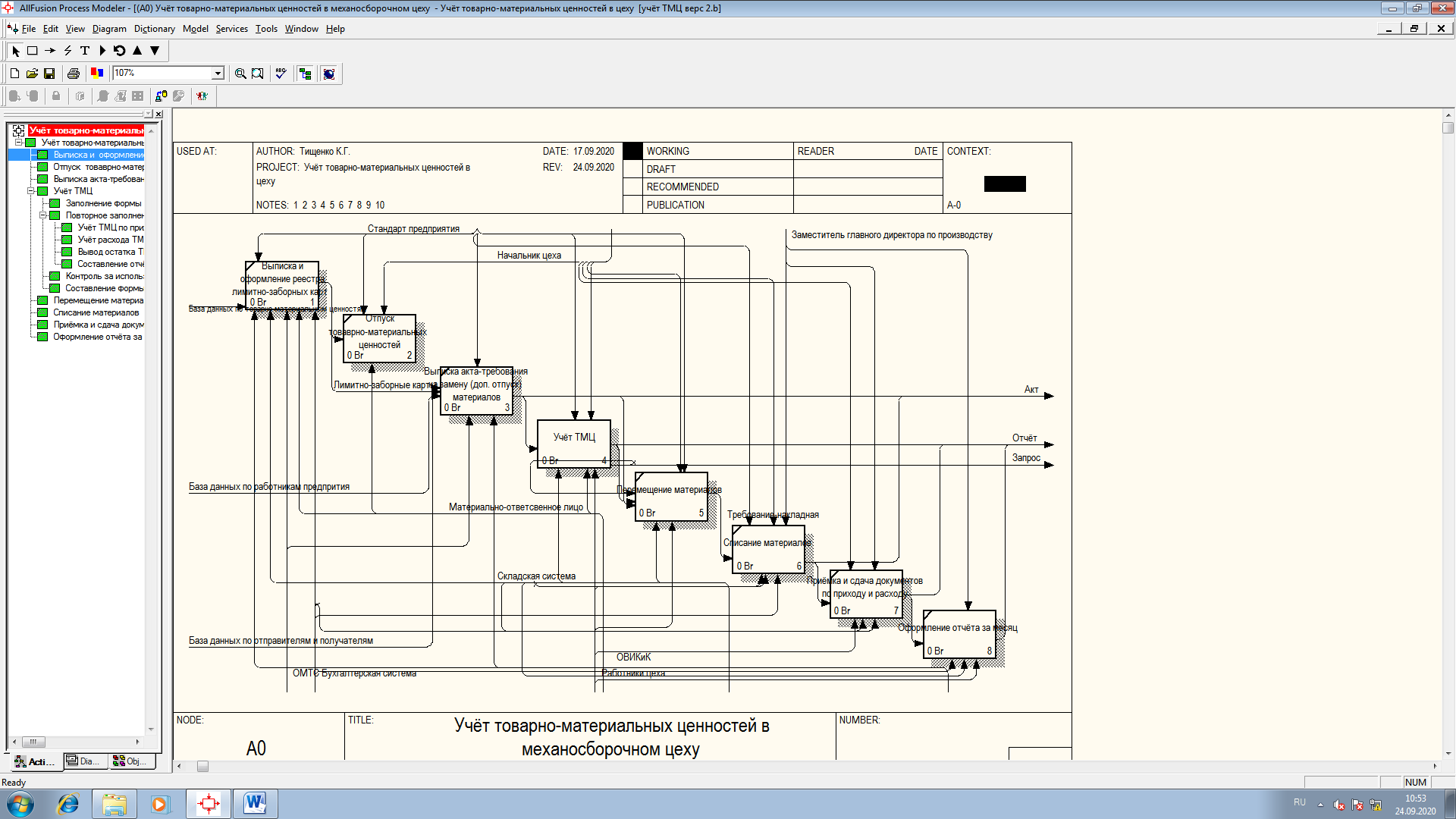


Рисунок 3 – Декомпозиция процесса «Определение статуса задачи»

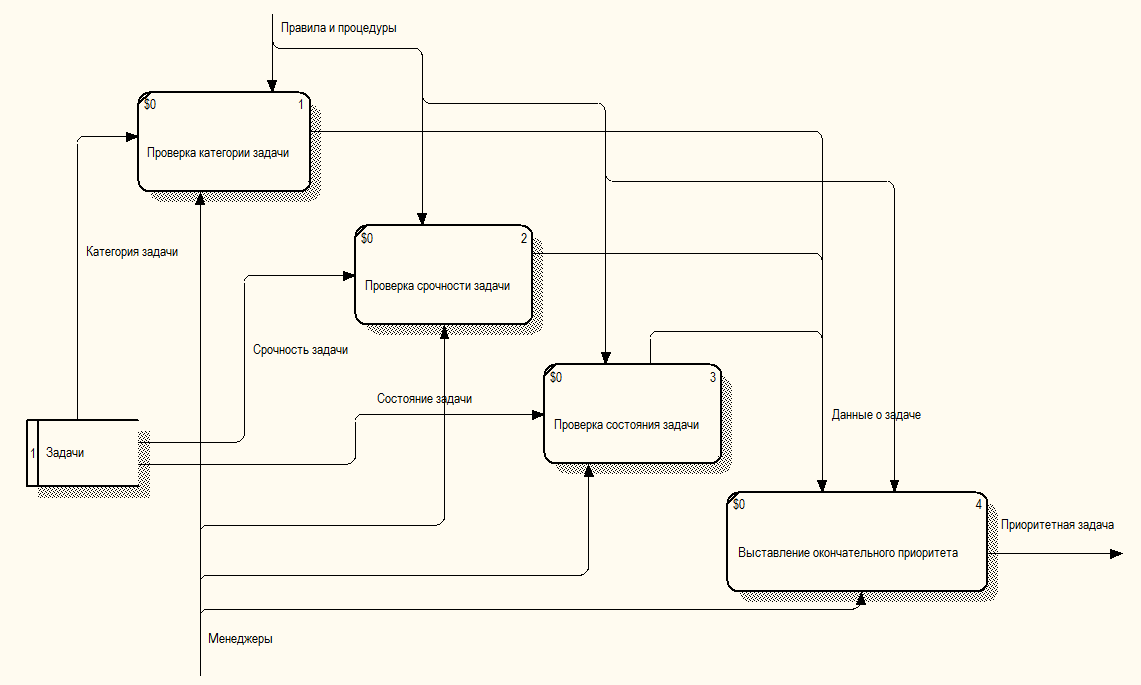


Рисунок 4 – Декомпозиция процесса «Решение задачи»

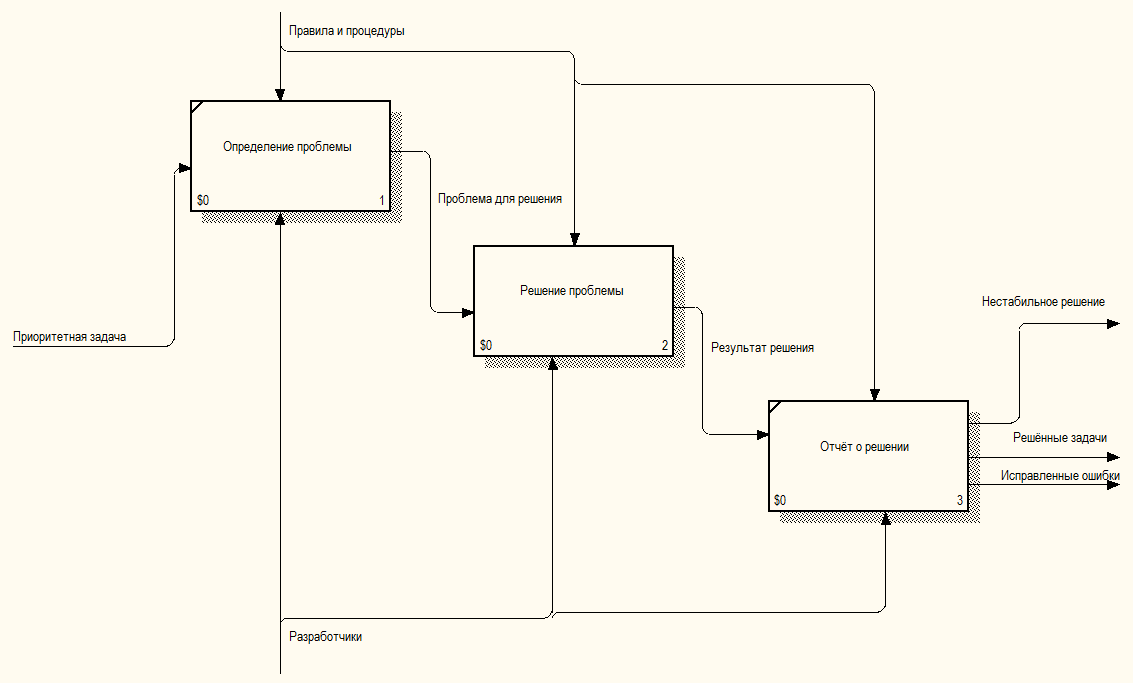


Рисунок 5 – Декомпозиция процесса «Тестирование задачи»

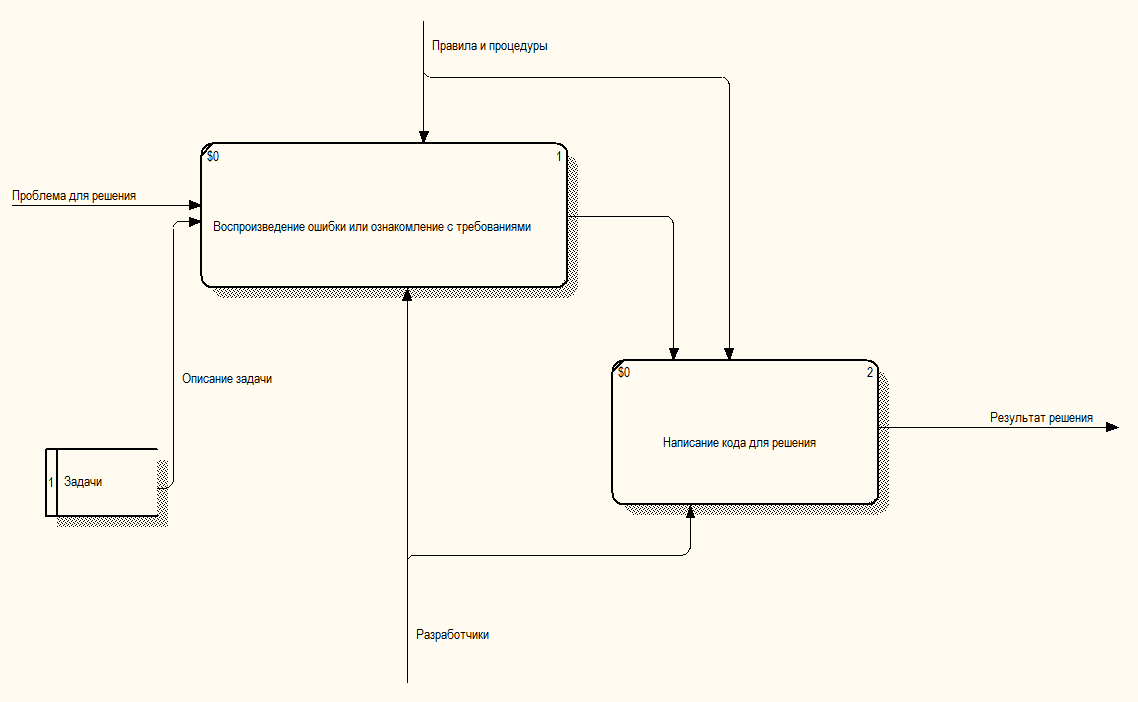


Рисунок 6 – Декомпозиция процесса «Внедрение решённой задачи»

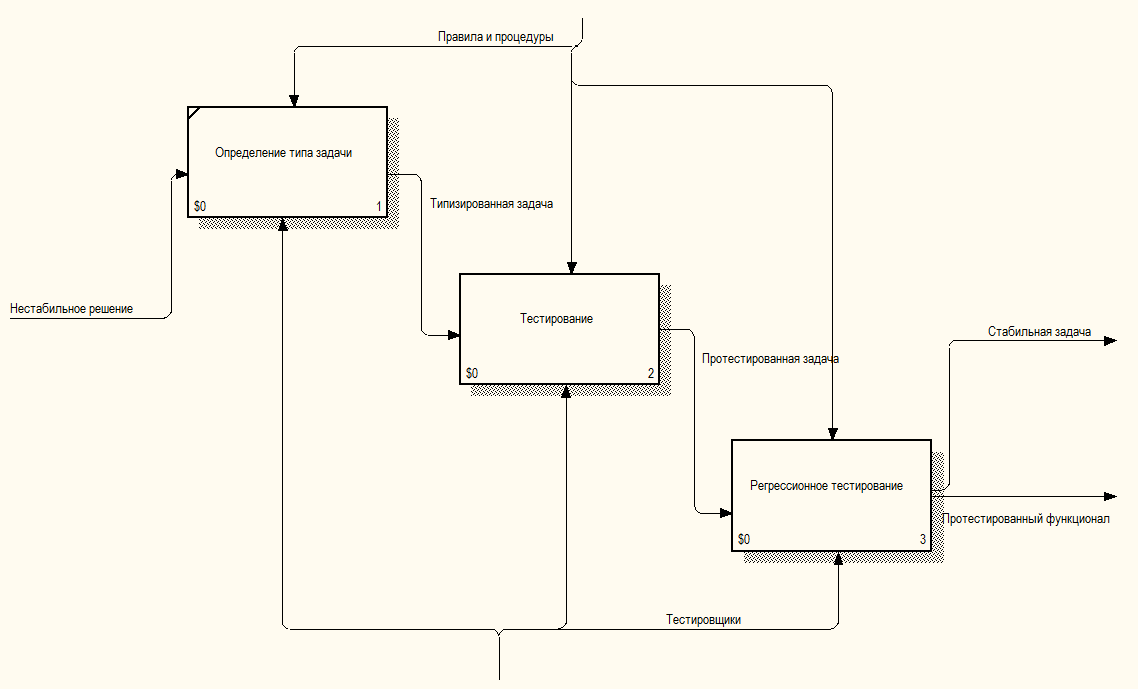


Рисунок 7 – Декомпозиция процесса «Выставление приоритета задачи»

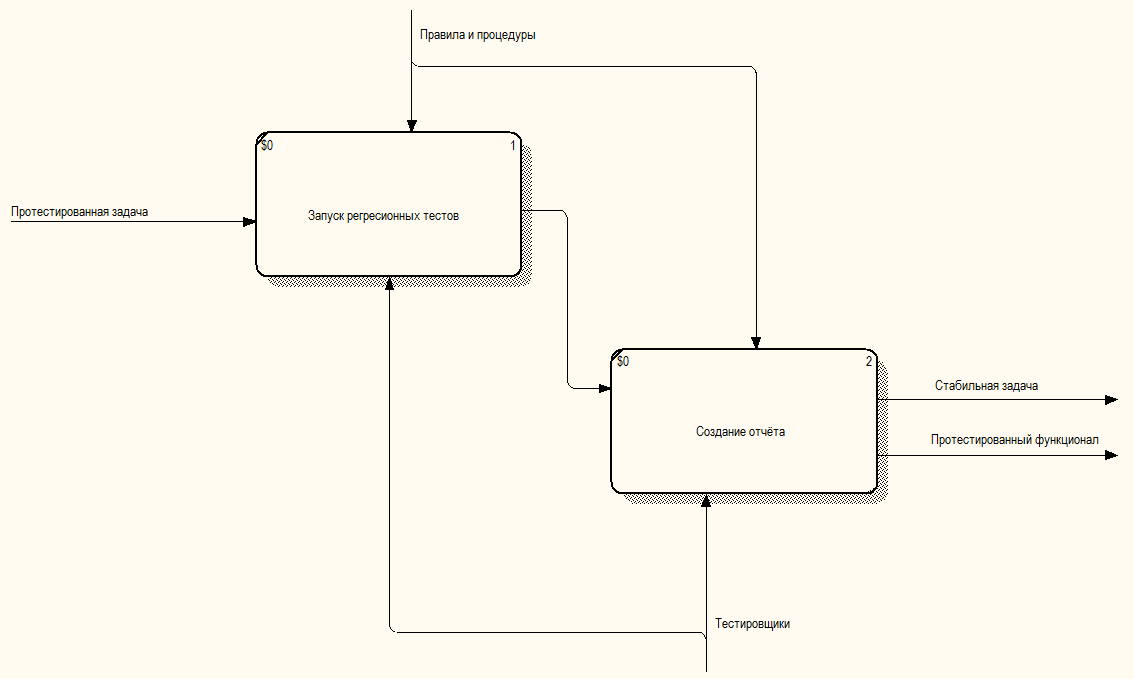


Рисунок 8 – Декомпозиция процесса «Решение проблемы»

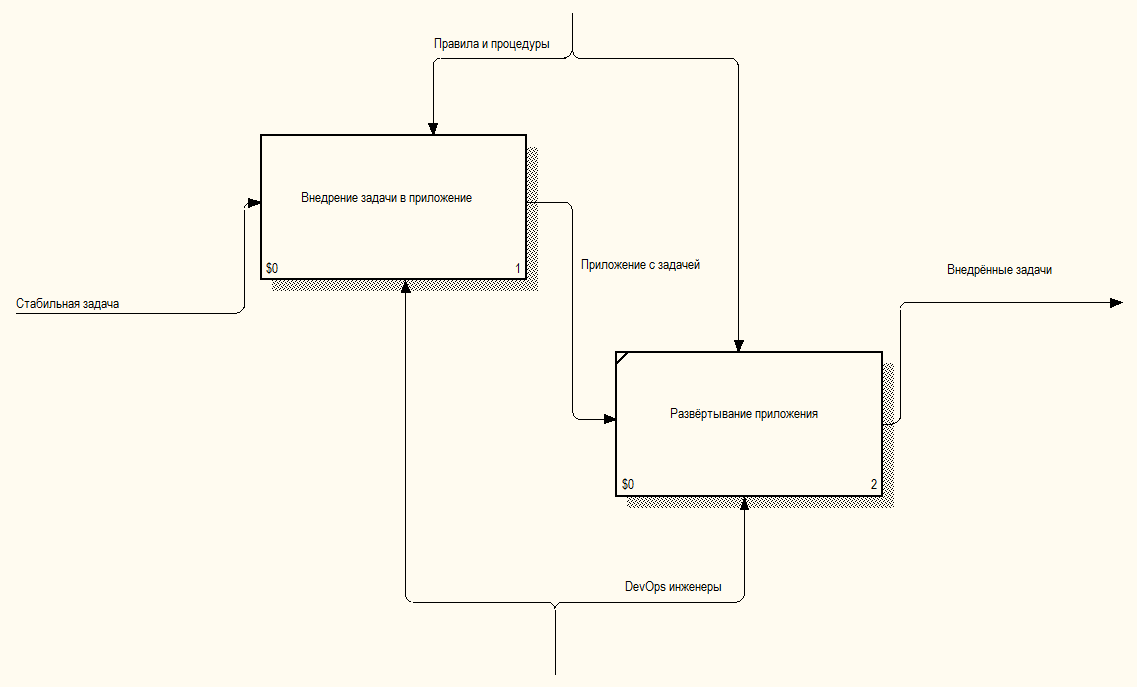


Рисунок 9 – Декомпозиция процесса «Внедрение задачи в приложение»

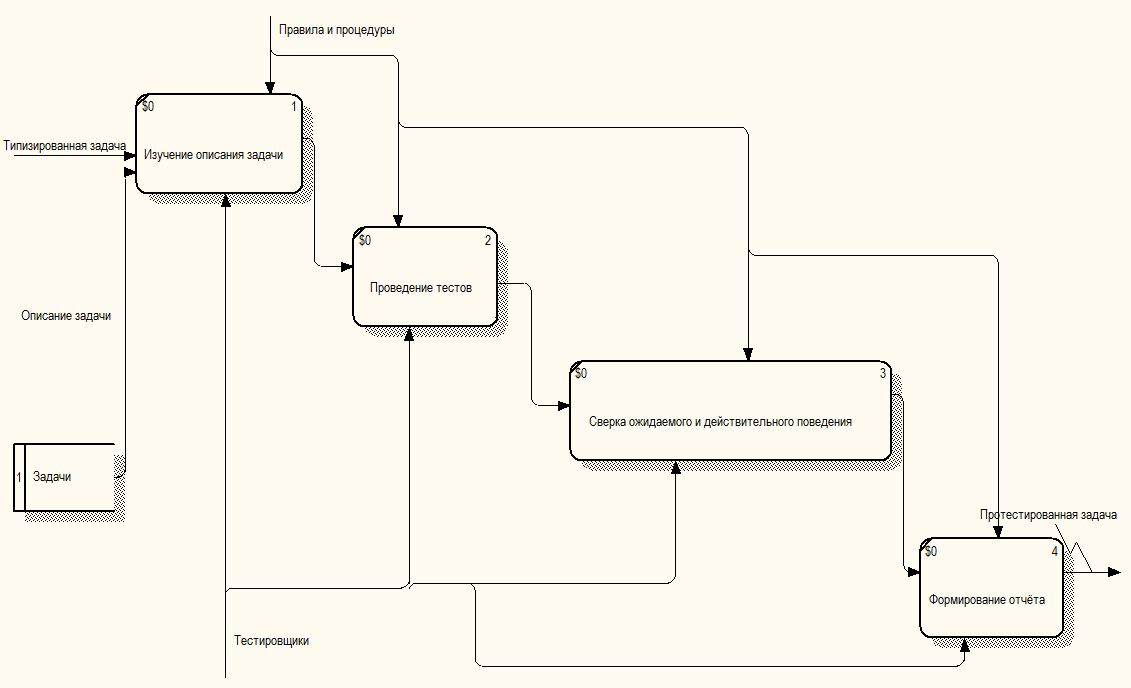


Рисунок 10 – Декомпозиция процесса «Выставление приоритета задачи»

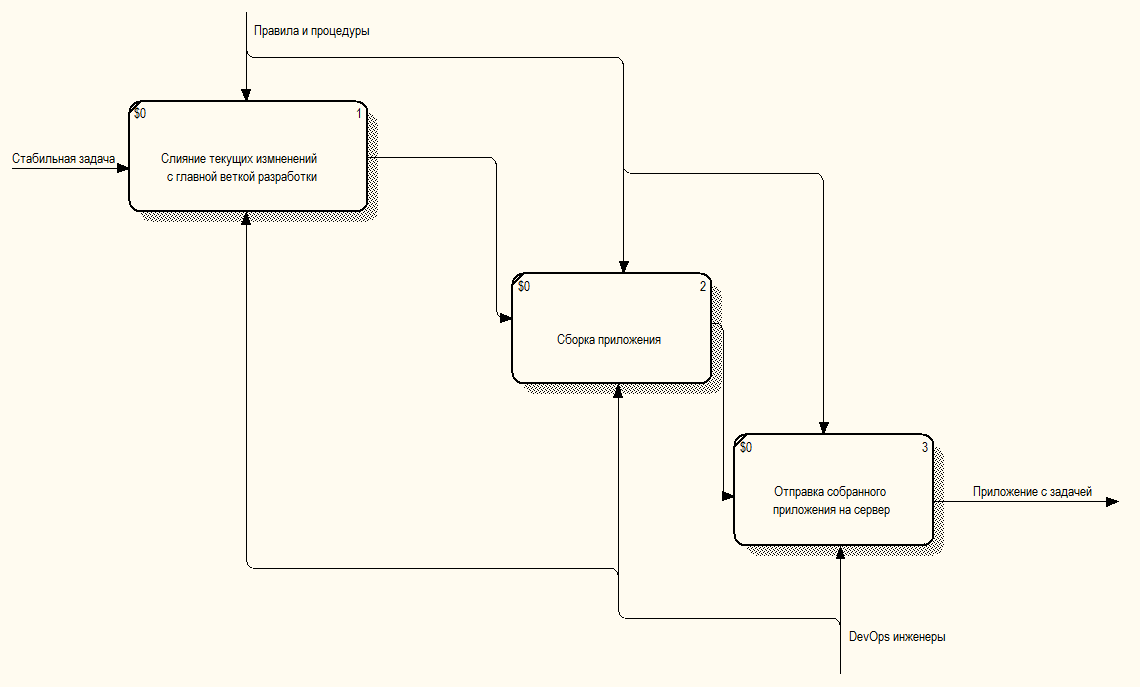
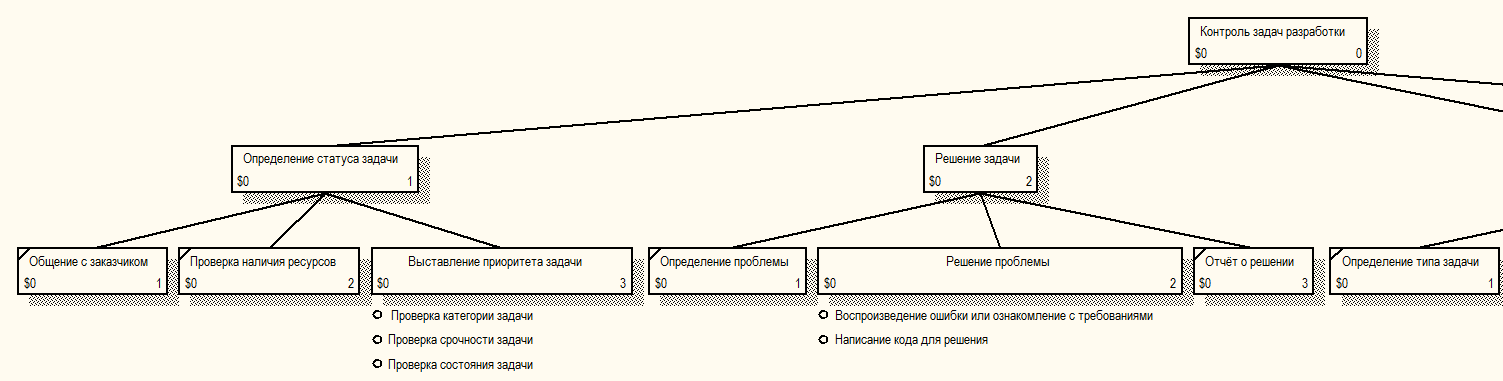


Рисунок 11 – Декомпозиция процесса «Регресионное тестирование»

2.1.2 Диаграмма дерева узлов

Диаграмма дерева узлов проектируемой базы данных представлена на рисунке 12.



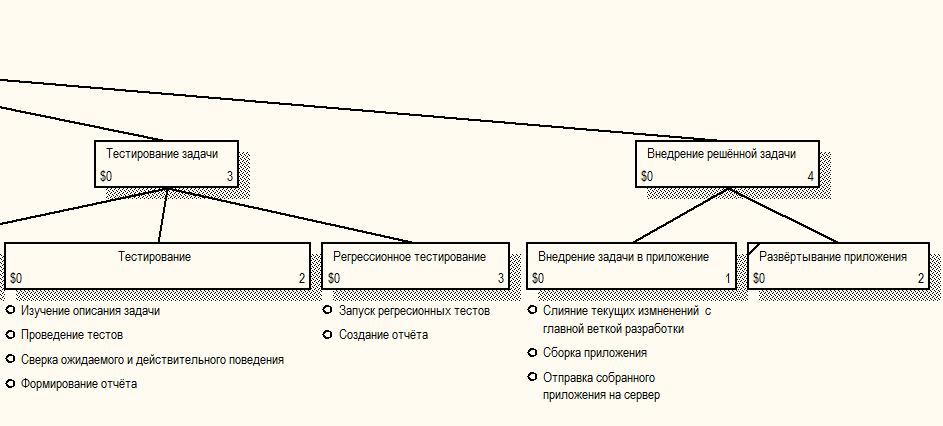


Рисунок 12– Диаграмма дерева узлов

2.2 Информационная модель

2.2.1 Идентификация сущностей и связей. ER-диаграмма логического уровня

Erwin имеет два уровня представления модели – логический и физический. Логический уровень – это абстрактный взгляд на данные. Объекты модели, представляемые на нём, называются сущностями и атрибутами. Логическая модель данных является универсальной, т.к. не зависит от конкретной СУБД.

Для отображения информационной модели рассматриваемого процесса на логической модели используются следующие сущности:

* «Workers». Данная сущность хранит в себе информацию о сотрудниках предприятия;
* «Organizations». Данная сущность содержит в информацию об организациях;
* «Cheta». Данная сущность содержит информацию о бухгалтерских счетах;
* «AdvanceReports». Данная сущность содержит информацию об авансовых отчётах;
* «RecordMF». Данная сущность содержит информацию об объектах основных средств;
* «WriteOffMF». Данная сущность содержит информацию о списании объектов основных средств;
* «RecordStocks». Данная сущность содержит информацию о запасах;
* «WriteOffS». Данная сущность содержит информацию о списании запасов;
* «RecordPayment». Данная сущность содержит информацию о расчётах с поставщиками и подрядчиками;
* «RecordOperation». Данная сущность содержит информацию о банковских операциях.

Связь между сущностями Workers и AdvanceReports не идентифицирующая, т.к. авансовый отчет может существовать без имени подотчетного лица. Тип связи один-ко-многим, т.к. у нескольких отчетов может быть одно подотчетное лицо, у одного отчета может быть только одно подотчетное лицо.

Связь между сущностями Cheta и AdvanceReports не идентифицирующая, т.к. авансовый отчет может существовать без номера счета. Тип связи один-ко-многим, т.к. у нескольких отчетов может быть один номер счета, у одного отчета может быть только один номер счета.

Связь между сущностями Workers и RecordMF не идентифицирующая, т.к. объект основных средств может существовать без имени материально-ответственного лица. Тип связи один-ко-многим, т.к. у нескольких объектов основных средств может быть одно ответственное лицо, у одного объекта может быть только одно ответственное лицо.

Связь между сущностями Cheta и RecordMF не идентифицирующая, т.к. объект основных средств может существовать без номера счета. Тип связи один-ко-многим, т.к. у нескольких объектов основных средств может быть один номер счета, у одного объекта может быть только один номер счета.

Связь между сущностями WriteOffMF и RecordMF не идентифицирующая, т.к. списание объекта основных средств может осуществляться без id объекта основных средств. Тип связи один-ко-многим, т.к. у нескольких актов списания основных средств может быть один id объекта основных средств, у акта списания может быть только один id объекта основных средств.

Связь между сущностями Cheta и RecordOperation не идентифицирующая, т.к. банковская операция может существовать без номера счета. Тип связи один-ко-многим, т.к. у нескольких банковских операций может быть один номер счета, у одной операции может быть только один номер счета.

Связь между сущностями Cheta и RecordPayment не идентифицирующая, т.к. расчет может существовать без номера счета. Тип связи один-ко-многим, т.к. у нескольких расчетов может быть один номер счета, у одного расчета может быть только один номер счета.

Связь между сущностями Organizations и RecordPayment не идентифицирующая, т.к. расчет может существовать без наименования организации. Тип связи один-ко-многим, т.к. у нескольких расчетов может быть одно наименование организации, у одного расчета может быть только одно наименование организации.

Связь между сущностями Cheta и RecordStocks не идентифицирующая, т.к. учет может существовать без номера счета. Тип связи один-ко-многим, т.к. у нескольких учетов может быть один номер счета, у одного учета может быть только один номер счета.

Связь между сущностями Organizations и RecordStocks идентифицирующая, т.к. учет не может существовать без ответственного лица. Тип связи многие-ко-многим, т.к. у одной организации может храниться несколько запасов, а один запас может хранится у нескольких организаций.

Связь между сущностями WriteOffS и RecordStocks не идентифицирующая, т.к. списание запаса может осуществляться без id запаса. Тип связи один-ко-многим, т.к. у нескольких актов списания запаса может быть один id запаса, у акта списания может быть только один id запаса.

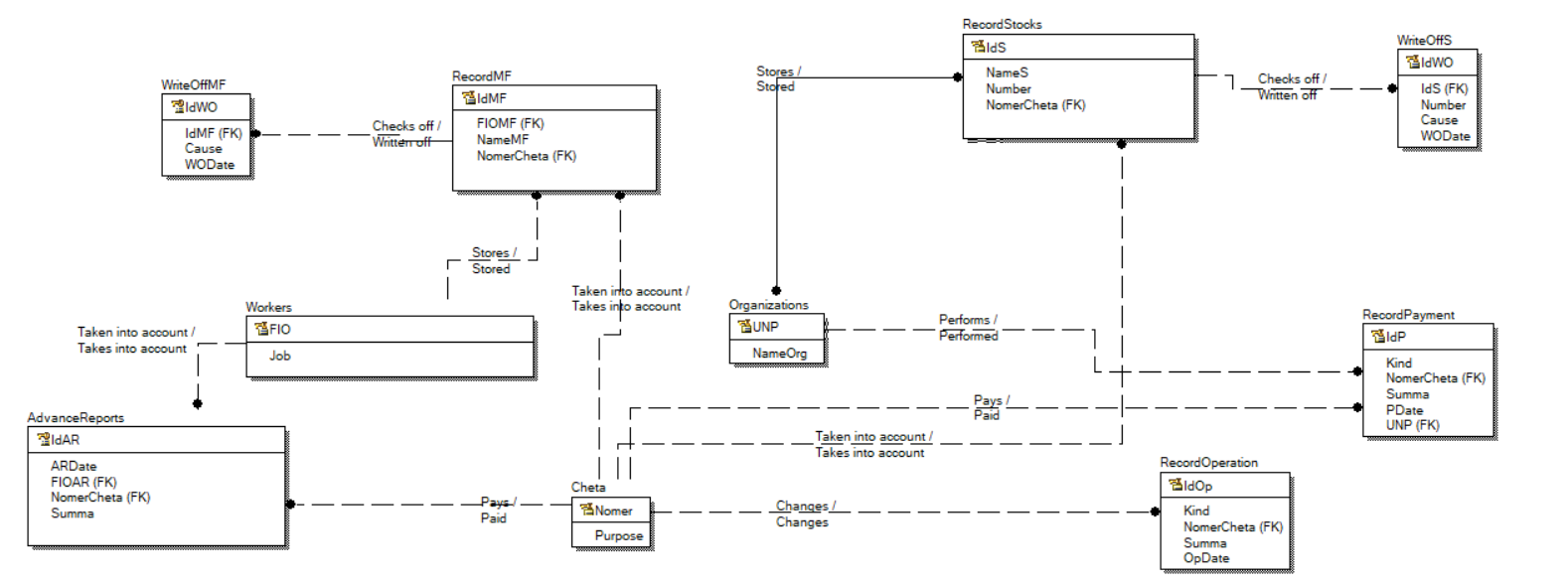


Рисунок 6 – ER-диаграмма логического уровня

2.2.2 ER-диаграмма физического уровня. Ограничения ссылочной целостности. Переопределение триггеров. Индексирование отношений.

Все связи в модели имеют одинаковую ссылочную целостность. Везде поддерживается каскадное удаление и обновление, т.к. при отсутствии или изменении некой информации в родительской сущности, она теряет свой смысл при использовании её в дочерних сущностях.

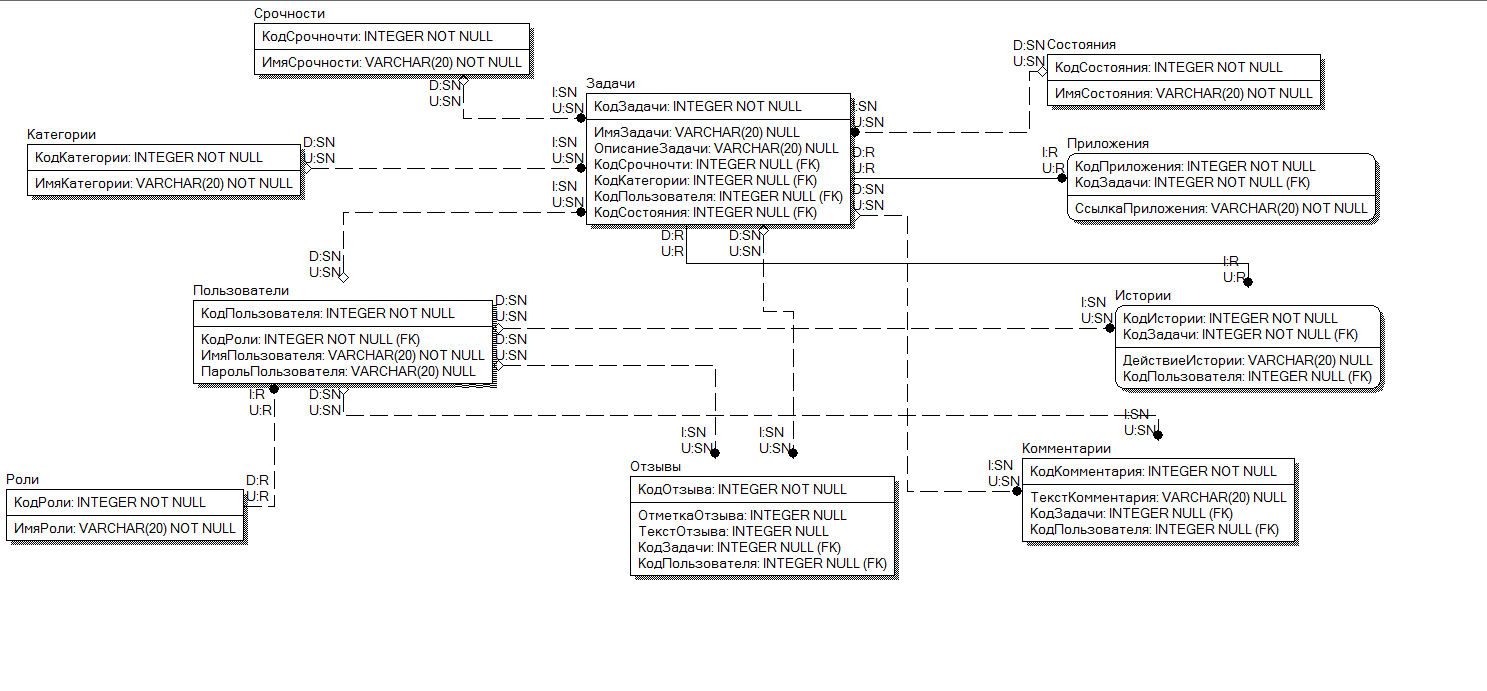


Рисунок 14 – ER-диаграмма физического уровня

Типы данных:

1. Рассмотрим таблицу «Tasks». Если взять поле «Name», то оно будет представлено строковым типом данных.
2. Рассмотрим таблицу «Tasks». Если взять поле «Description», то она представлена типом money.
3. Рассмотрим таблицу «Users». Если взять поле «RoleId», то оно представлено целочисленным типом данных (integer).
4. Рассмотрим таблицу «Timecards». Если взять поле «Date», то оно представлено типом datetime.

В базе данных триггеры определены для сущностей Tasks, Users, Comments. Данные триггеры на такие операции как INSERT, DELETE и UPDATE.

2.2.3 Определение представлений, хранимых процедур серверной компоненты

Для данной базы данных были созданы следующие представления:

1. AllTaskInfo – выбирает все поля из таблицы Tasks;
2. TaskBugInfo– выбирает все поля о задачах-багах, данные таблицы были связаны посредством JOIN;
3. NewTaskInfo– полностью выводится таблица Tasks, где указаны данные о новых задачах;
4. ActiveCriticalTaskInfo– полностью выводится таблица Tasks, где вместо идентификаторов из полей StateId и UrgencyId будет подставлено значение из связной таблицы.
5. TaskWithFixDescriptionInfo– выводит информацию о задачах, заменяя их описание на краткое описание в виде имени задачи;
6. TaskWithUserRoleInfo– выводит информацию о задачах и с помощью JOIN показывает роли пользователей, создавших их;
7. UserDevelopersInfo– выводится таблица Users, где указаны все разработчики из команды;
8. UserWithFeedbackInfo– полностью выводится таблица Users, где будут указаны оставленныи ими фидбеки из таблицы Feedbacks.
9. FeedbackWithRateBetween3And6Info– Выводи задачи, рейтинг которых находится между 3 и 5;
10. AttachementsWithHttpsReferenceInfo– выводит информацию о приложениях, которые являются ссылками с http;
11. UsersIdInfo– выводит все id пользователей;
12. FeedbackTextOnlyInfo – Выводит только текст определённого фидбека из таблицы Feedbacks;
13. UpdateUsers – обновляет пользователя по Id;
14. UserWithFeedbackInfo – выводятся данные об отзывах из таблицы Feedbacks и о пользователях, которые их оставили;
15. DeleteUsers – Удаляет пользователя по переданному Id.

2.3 Верификация спроектированной логической модели

Activity Name: { NO ACTIVITY }

Arrow Name: DevOps инженеры

Entity Name: Пользователи

Arrow Name: Категория задачи

Entity Name: Категории

Attribute Name: ИмяКатегории

Arrow Name: Менеджеры

Entity Name: Пользователи

Arrow Name: Новые задачи

Entity Name: Задачи

Entity Name: Категории

Entity Name: Приложения

Entity Name: Состояния

Entity Name: Срочности

Arrow Name: Разработчики

Entity Name: Пользователи

Arrow Name: Состояние задачи

Entity Name: Состояния

Attribute Name: ИмяСостояния

Arrow Name: Срочность задачи

Entity Name: Срочности

Attribute Name: ИмяСрочности

Arrow Name: Тестировщики

Entity Name: Пользователи

3 Реализация системы

3.1 SQL-определения регламентированных запросов

AllTaskInfo выбирает информацию о задачах

GO

CREATE VIEW AllTaskInfo

AS

SELECT Id, Name, Description

FROM Tasks

TaskBugInfo выбирает информацию о багах

GO

CREATE VIEW TaskBugInfo

AS

SELECT Id, Name, Description

FROM Tasks

WHERE CategoryId = 1;

NewTaskInfo выбирает информацию о новых задачах

GO

CREATE VIEW NewTaskInfo

AS

SELECT Id, Name, Description

FROM Tasks

WHERE StateId = 1;

ActiveCriticalTaskInfo выбирает информацию об активных задачах

GO

CREATE VIEW ActiveCriticalTaskInfo

AS

SELECT Id, Name, Description

FROM Tasks

WHERE StateId = 2 AND UrgencyId = 5;

FullAllTaskInfo выбирает полную информацию обо всех задачах

GO

CREATE VIEW FullAllTaskInfo

AS

SELECT t.Id as TaskId, t.Name, Description,

u.Name as Urgency, s.Name as State, c.Name as Category

FROM Tasks t

LEFT JOIN Urgencies u ON t.UrgencyId = u.Id

LEFt JOIN States s ON t.StateId = s.Id

LEFT JOIN Categories c ON t.CategoryId = c.Id;

TaskWithFixDescriptionInfo выбирает описание задач

GO

CREATE VIEW TaskWithFixDescriptionInfo

AS

SELECT Id, Name, Description

FROM Tasks

WHERE Description LIKE 'Fix%';

TaskWithUserRoleInfo выбирает задачи, с ролью пользователя, их создавшего

GO

CREATE VIEW TaskWithUserRoleInfo

AS

SELECT t.Name, Description, r.Name as UserRole

FROM Tasks t

LEFT JOIN Users u ON t.UserId = u.Id

LEFT JOIN Roles r ON u.RoleId = r.Id

WHERE CategoryId = 1;

UserDevelopersInfo выводит информацию о пользователях с ролью ‘Разработчик’

GO

CREATE VIEW UserDevelopersInfo

AS

SELECT Id, Name

FROM Users

WHERE RoleId = 2;

UserWithFeedbackInfo выводит информацию о пользователях и комментариях, которые они оставляли

GO

CREATE VIEW UserWithFeedbackInfo

AS

SELECT Name, f.Text

FROM Users u

RIGHT JOIN Feedbacks f ON u.Id = f.UserId;

FeedbackWithRateBetween3And6Info выводит отзывы с оценкой 4 и 5

GO

CREATE VIEW FeedbackWithRateBetween3And6Info

AS

SELECT Text, Rate, UserId

FROM Feedbacks

WHERE Rate BETWEEN 3 AND 6;

AttachementsWithHttpsReferenceInfo выводит все приложения со ссылкой про протоколу HTTPS

GO

CREATE VIEW AttachementsWithHttpsReferenceInfo

AS

SELECT Id, Reference

FROM Attachements

WHERE Reference LIKE 'https://%';

UserNotManagersInfo выводит всех пользователей, кроме менеджеров

GO

CREATE VIEW UserNotManagersInfo

AS

SELECT u.Id, u.Name, r.Name as Role

FROM Users u

LEFT JOIN Roles r ON u.RoleId = r.Id

WHERE RoleId != 3;

UsersIdInfo выводит все id пользователей

GO

CREATE VIEW UsersIdInfo

AS

SELECT Id

FROM Users;

HistoriesMadeByTestersInfo информация об историях произведённых пользователем

GO

CREATE VIEW HistoriesMadeByTestersInfo

AS

SELECT h.Id, h.Action, h.TaskId

FROM Users u

LEFT JOIN Histories h ON h.UserId = u.Id

WHERE u.RoleId = 4;

FeedbackTextOnlyInfo выводит информацию об отзывах в текстовом формате

GO

CREATE VIEW FeedbackTextOnlyInfo

AS

SELECT Text

FROM Feedbacks;

3.2 SQL-определения триггеров

Tasks\_INSERT триггер понижающий срочность задачи при вставке

GO

CREATE TRIGGER Tasks\_INSERT

ON dbo.Tasks

AFTER INSERT

AS

UPDATE dbo.Tasks

SET UrgencyId = UrgencyId - 1

WHERE Id = (SELECT Id FROM inserted);

Attachements\_DELETE триггер сробатывающий при удалении приложения

GO

CREATE TRIGGER Attachements\_DELETE

ON dbo.Tasks

AFTER INSERT

AS

UPDATE dbo.Tasks

SET UrgencyId = UrgencyId - 1

WHERE Id = (SELECT Id FROM inserted);

Tasks\_DELETE выводит имя удалённой задачи

GO

CREATE TRIGGER Tasks\_DELETE

ON dbo.Tasks

AFTER DELETE

AS

SELECT Name FROM deleted;

Users\_DELETE выводит имя и роль удалённого пользователя

GO

CREATE TRIGGER Users\_DELETE

ON dbo.Users

AFTER DELETE

AS

SELECT Name + RoleId FROM deleted;

Users\_INSERT страбатывает при вставке юзера и обновляет его имя

GO

CREATE TRIGGER Users\_INSERT

ON dbo.Users

AFTER INSERT

AS

UPDATE dbo.Users

SET Name = Name + (SELECT r.Name FROM dbo.Roles r WHERE r.Id = RoleId)

WHERE Id = (SELECT Id FROM inserted);

Триггер добавляющий дату в новую историю

GO

CREATE TRIGGER Histories\_INSERT

ON dbo.Histories

AFTER INSERT

AS

UPDATE dbo.Histories

SET Action = Action + GETDATE()

WHERE Id = (SELECT Id FROM inserted);

Feedbacks\_INSERT обновляет описание отзыва при вставке

GO

CREATE TRIGGER Feedbacks\_INSERT

ON dbo.Feedbacks

AFTER INSERT

AS

UPDATE dbo.Feedbacks

SET Text = Text + ': ' + Rate

WHERE Id = (SELECT Id FROM inserted);

Feedbacks\_INSERT обновляет описание отзыва при изменении

GO

CREATE TRIGGER Feedbacks\_UPDATE

ON dbo.Feedbacks

AFTER INSERT

AS

UPDATE dbo.Feedbacks

SET Text = Text + ': ' + Rate

WHERE Id = (SELECT Id FROM inserted);

Добавляет дату создвния комментария при вставке

GO

CREATE TRIGGER Comments\_INSERT

ON dbo.Comments

AFTER INSERT

AS

UPDATE dbo.Comments

SET Text = Text + GETDATE()

WHERE Id = (SELECT Id FROM inserted)

Заменяет все HTTP протоколы на HTTPS - безопасные

GO

CREATE TRIGGER Attachements\_INSERT

ON dbo.Attachements

AFTER INSERT

AS

UPDATE dbo.Attachements

SET Reference = REPLACE(Reference, 'http://', 'https://')

WHERE Id = (SELECT Id FROM inserted) AND Reference LIKE 'http://%'

3.3 SQL-определения хранимых процедур

TaskBugInfo выводит информацию о задачах типа ‘баг’

GO

CREATE PROCEDURE TaskBugInfo

AS

SELECT Id, Name, Description

FROM Tasks

WHERE CategoryId = 1;

Выводит информацию о задачах и о пользователях, которое создали эти задачи

GO

CREATE PROCEDURE TaskAndUsersInfo

AS

SELECT Id, t.Name, Description, u.Name

FROM Tasks t

LEFT JOIN Users u

WHERE CategoryId = 1 OR u.RoleId = 0;

Выводит полное описание задачи, получая данные из связных таблиц

GO

CREATE PROCEDURE FullAllTaskInfo

AS

SELECT t.Id as TaskId, t.Name, Description,

u.Name as Urgency, s.Name as State, c.Name as Category

FROM Tasks t

LEFT JOIN Urgencies u ON t.UrgencyId = u.Id

LEFt JOIN States s ON t.StateId = s.Id

LEFT JOIN Categories c ON t.CategoryId = c.Id;

Показывает задачи и роли пользователей, их создавших

GO

CREATE PROCEDURE TaskWithUserRoleInfo

AS

SELECT t.Name, Description, r.Name as UserRole

FROM Tasks t

LEFT JOIN Users u ON t.UserId = u.Id

LEFT JOIN Roles r ON u.RoleId = r.Id

WHERE CategoryId = 1;

UserDevelopersInfo производит выборку пользователей-разработчиков

GO

CREATE PROCEDURE UserDevelopersInfo

AS

SELECT Id, Name

FROM Users

WHERE RoleId = 2;

Выводит все HTTPS ссылки в приложениях

GO

CREATE PROCEDURE AttachementsWithHttpsReferenceInfo

AS

SELECT Id, Reference

FROM Attachements

WHERE Reference LIKE 'https://%';

Выводит все id пользователей

GO

CREATE PROCEDURE UsersIdInfo

AS

SELECT Id

FROM Users;

Показывает первые буквы имён пользователей

GO

CREATE PROCEDURE UsersFirstLetter

AS

SELECT

SUBSTRING(Name, 1, 1)

FROM Users;

Показывает истории только тестировщиков

GO

CREATE PROCEDURE HistoriesMadeByTestersInfo

AS

SELECT h.Id, h.Action, h.TaskId

FROM Users u

LEFT JOIN Histories h ON h.UserId = u.Id

WHERE u.RoleId = 4;

Выводит только тексты отзывов

GO

CREATE PROCEDURE FeedbackTextOnlyInfo

AS

SELECT Text

FROM Feedbacks;

Обновляет отметку отзыва

GO

CREATE PROCEDURE UpdateFeedbacks

AS

UPDATE Feedbacks SET Rate = 5

WHERE Rate = 4;

UpdateUsers обновляет пользователей, меняя их пароль

GO

CREATE PROCEDURE UpdateUsers

AS

UPDATE Users SET Name = ‘TestUser’

WHERE Password = ‘qwerty’;

UpdateUsersWithEptyPassword обновляет пользователей

GO

CREATE PROCEDURE UpdateUsersWithEptyPassword

AS

UPDATE Users SET Password = '1'

WHERE Password = '';

Удаляет задачу по Id

GO

CREATE PROCEDURE DeleteTasks

AS

DELETE FROM Tasks

WHERE StateId = 5;

DeleteDoneTasks удаляет все сделанные задачи

GO

CREATE PROCEDURE DeleteDoneTasks

AS

DELETE FROM Tasks

WHERE StateId = 5 AND Name=’Test’;

Удаляет задачи по определённому состоянию

GO

CREATE PROCEDURE DeleteSpecificTasks

AS

DELETE FROM Tasks

WHERE StateId = 5 OR StateId = 4;

DeleteUsersTesters удаляет из базы всех тестировщиков

GO

CREATE PROCEDURE DeleteUsersTesters

AS

DELETE FROM Users

WHERE RoleId = 3;

Чистит всех пользователей с именем начинающемся на ‘A’

GO

CREATE PROCEDURE DeleteUsers

AS

DELETE FROM Users

WHERE (SELECT Name AS Role

FROM Roles r WHERE r.Id = Id) LIKE 'A%';

Выводит все пользователей, и составленные ими отзывы

GO

CREATE PROCEDURE UserWithFeedbackInfo

AS

SELECT Name, f.Text

FROM Users u

RIGHT JOIN Feedbacks f ON u.Id = f.UserId;

Показывает все отзывы с отметкой между 3 и 5

GO

CREATE VIEW FeedbackWithRateBetween3And6Info

AS

SELECT Text, Rate, UserId

FROM Feedbacks

WHERE Rate BETWEEN 3 AND 6;

3.4 SQL-определения курсоров

Cursor1 для выборки имён пользователей

GO

DECLARE Cursor1 CURSOR

FOR

SELECT Name FROM Users

OPEN Cursor1

FETCH Cursor1

ClOSE Cursor1

DEALLOCATE Cursor1

Cursor2 используется для вывода первых букв всех членов команды(полезно для вывода аватарок с первой буквой имени или фамилии)

GO

DECLARE Cursor2 CURSOR

FOR

SELECT Name, Id FROM Users

DECLARE @Name char(50), @Id int

OPEN Cursor2

FETCH Cursor2 INTO @Name, @Id

PRINT RTRIM(@Name)

ClOSE Cursor2

DEALLOCATE Cursor2

Cursor3 используется для вывода краткой статистики количества различных членов команды

GO

DECLARE Cursor3 CURSOR

LOCAL

FAST\_FORWARD

FOR

SELECT Name FROM Users

DECLARE @Name char(50), @count int

OPEN Cursor3

SET @count = 1

WHILE @count <= 15

BEGIN

FETCH Cursor3 INTO @Name

PRINT RTRIM(@count) + ': ' + RTRIM(@Name)

SET @count = @count + 1

END

ClOSE Cursor3

DEALLOCATE Cursor3

Cursor4 используется для выборки паролей и изменении пустых паролей

GO

DECLARE Cursor4 CURSOR

FOR

SELECT Password FROM Users

OPEN Cursor4

FETCH Cursor4

UPDATE Users SET Password = ''

WHERE CURRENT OF Cursor4

SELECT Password FROM Users WHERE Id = 1

ClOSE Cursor4

DEALLOCATE Cursor4

Cursor5 используется для выборки имён пользователей и удаления пробелов в них

GO

DECLARE Cursor5 CURSOR

LOCAL

KEYSET

FOR

SELECT Name FROM Users

DECLARE @Name char(50), @SecondName char(50)

OPEN Cursor5

FETCH FIRST FROM Cursor5 INTO @Name

FETCH ABSOLUTE 5 FROM Cursor5 INTO @SecondName

PRINT RTRIM(@Name) + ' meets ' + RTRIM(@SecondName)

ClOSE Cursor5

DEALLOCATE Cursor5

3.5 Распределение прав доступа пользователей

CREATE LOGIN Guest

WITH PASSWORD = 'guest01'

CREATE LOGIN Developer

WITH PASSWORD = 'dev01'

CREATE LOGIN Manager

WITH PASSWORD = 'manager'

CREATE LOGIN Tester

WITH PASSWORD = ‘qwerty’

CREATE LOGIN DevOps

WITH PASSWORD = 'DevooooPs'

CREATE USER Guest1 FOR LOGIN Guest

CREATE USER Developer1 FOR LOGIN Developer

CREATE USER Manager1 FOR LOGIN Manager

CREATE USER Tester1 FOR LOGIN Tester

CREATE USER DevOps1 FOR LOGIN DevOps

Пользователю Manager1 будут предоставленны все права на базу данных JustInMind, так же неявно ему предоставляется возможность самому предоставлять разрешения для данного объекта

USE JustInMind

GRANT CONTROL ON DATABASE JustInMind

TO Manager1

Пользователям DevOps1 и Developer1 предоставляются разрешение на создание таблиц и хранимых процедур, выполнение хранимых процедур, разрешение извлекать данные из таблицы или представления, разрешение модифицировать существующие данные в таблице или представлении, разрешение на вставку новых данных в таблицы, а также право самому предоставлять это разрешение другим пользователям базы данных.

USE JustInMind

GRANT CREATE TABLE, CREATE PROCEDURE, EXECUTE, DELETE, SELECT, INSERT, UPDATE

TO DevOps1, Developer1

WITH GRANT OPTION

Пользователю Tester1 запрещется производить операции вставки, удаления и обновления в таблице User, а так же разрешения, запрещенные пользователю Tester1, будут также запрещены пользователям, которым он их предоставил

USE JustInMind

DENY INSERT ON [JustInMind].[dbo].[User] TO Tester1 CASCADE

DENY UPDATE ON [JustInMind].[dbo].[User] TO Tester1 CASCADE

DENY DELETE ON [JustInMind].[dbo].[User] TO Tester1 CASCADE

Пользователю Tester1 предоставлены права на разрешение извлекать данные из таблицы или представления, а так же выполнять хранимые процедуры

USE JustInMind

GRANT SELECT, EXECUTE

TO GuestWithMaxRight

Пользователю Guest1 запрещается извлекать из таблицы User поле Password, так же запрещается выполнять хранимые процедуры.

USE JustInMind

DENY SELECT ON [JustInMind].[dbo].[User](Password) TO Guest1

DENY EXECUTE TO Guest1

4 Результаты тестирования информационной системы

Работа в приложении начинается с логина в приложении, если вы ещё не залогинились. В этом случае приложение вас предупредит об этом и переадресует на страницу Login

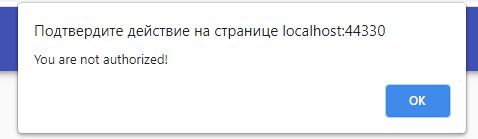


Рисунок 15 – Окно предупреждения о необходимости входа

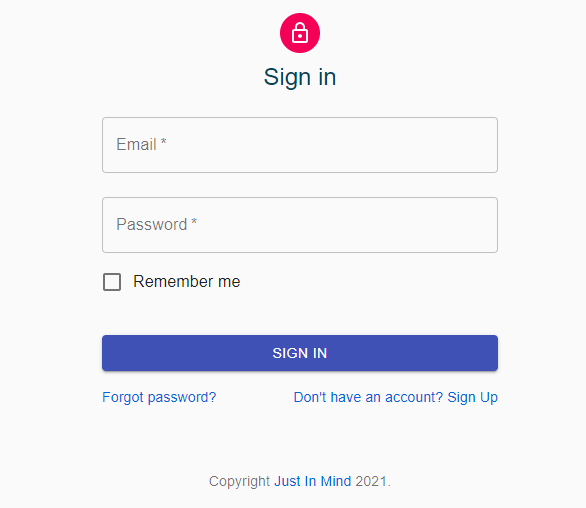


Рисунок 16 – Окно для логина

При вводе некорректных данных, появится всплывающее окошко. Подобного рода сообщения появляются на любое некорректное действие пользователя.

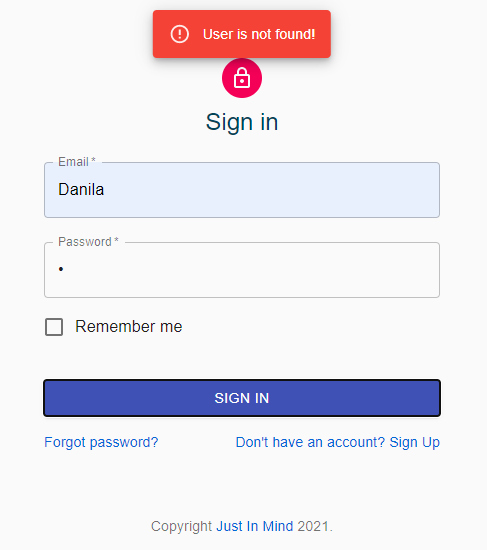


Рисунок 17 – Окно неправильного ввода данных при логине

При вводе корректных данных происходит переадресация на основную часть приложения. На ней представлено меню управления проектом, текущие задачи и небольшой личный кабинет пользователя. При клике на кнопу личного кабинета всплывает окно, позволяющее разлогиниться. Меню предоставляет доступ к проекты и ко всем членам команды. Эта таблица предоставляет богатый функционал: удаление одного/множества пользователей, приглашение нового члена команды, удобный механизм сортировки и пагинации.

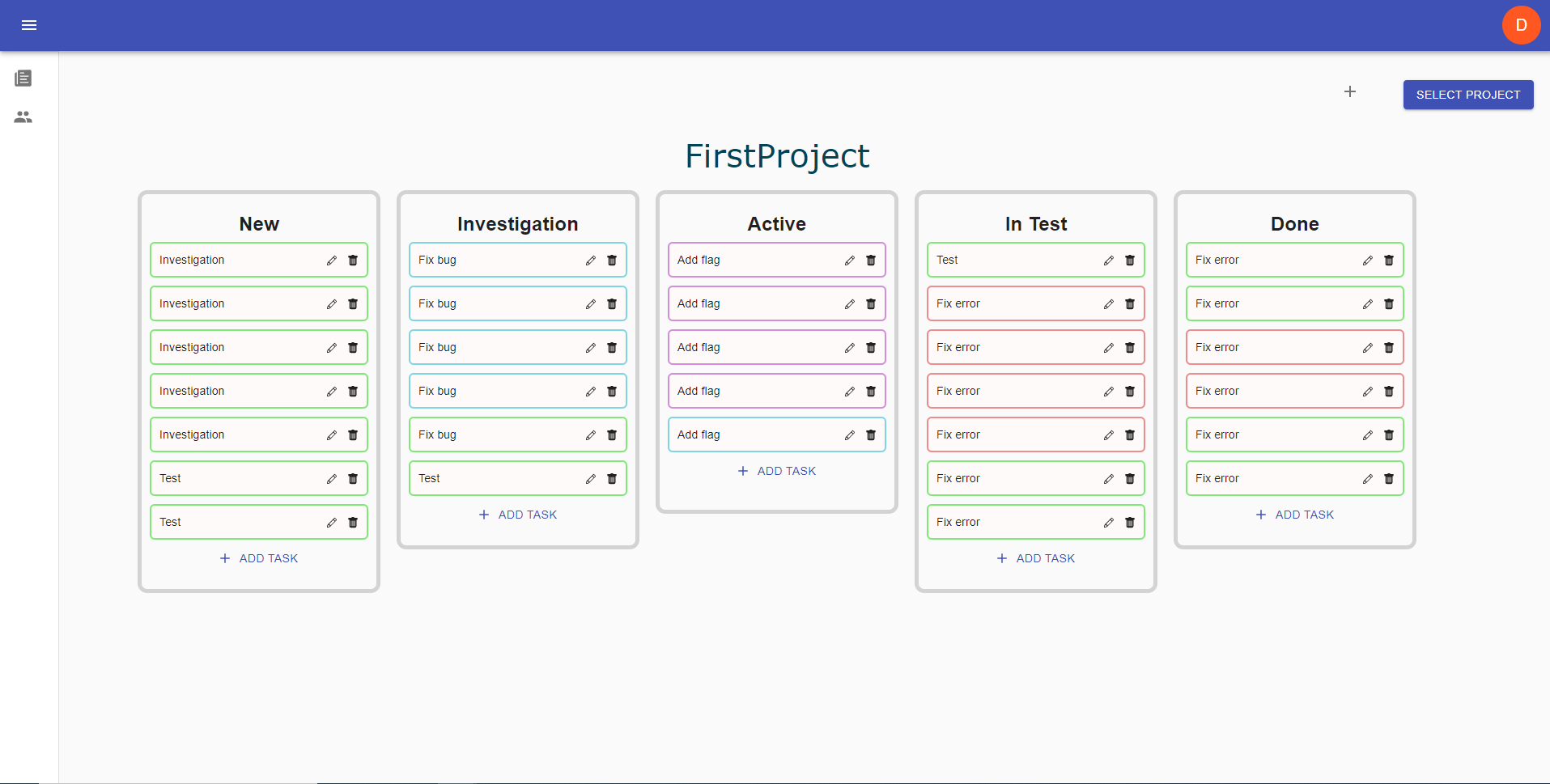


Рисунок 18 – Основное окно приложения



Рисунок 19 – Меню



Рисунок 20 – Кнопка личного кабинета



Рисунок 21 – Личный кабинет

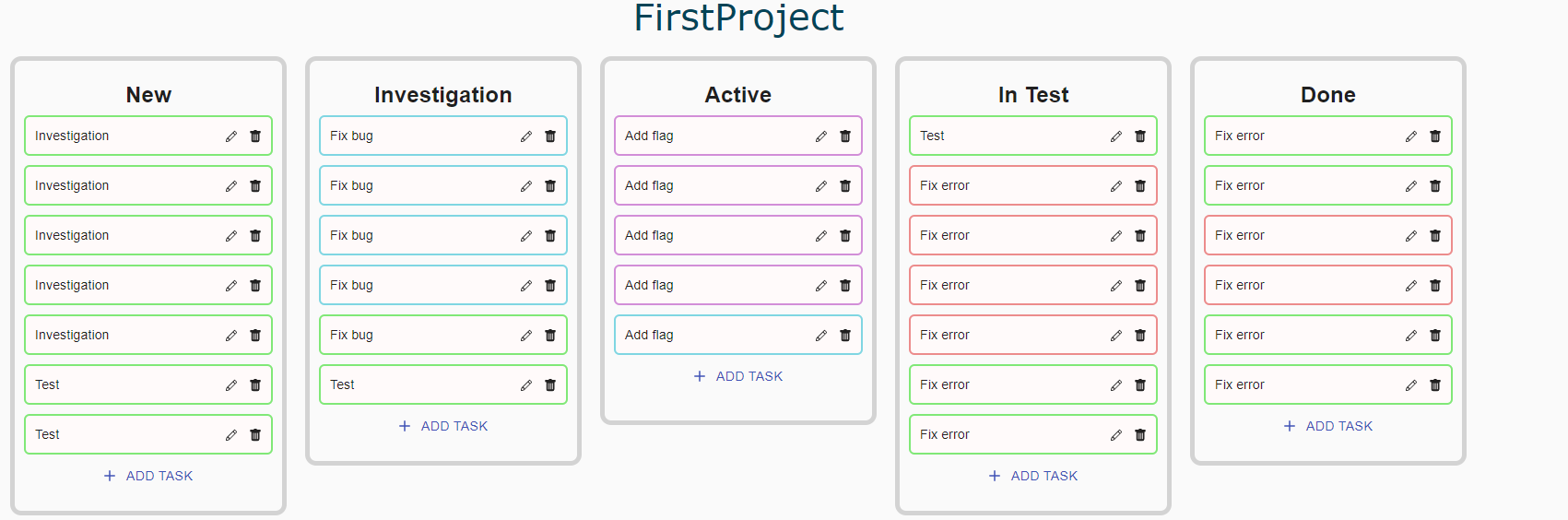


Рисунок 22 – Задачи проекта

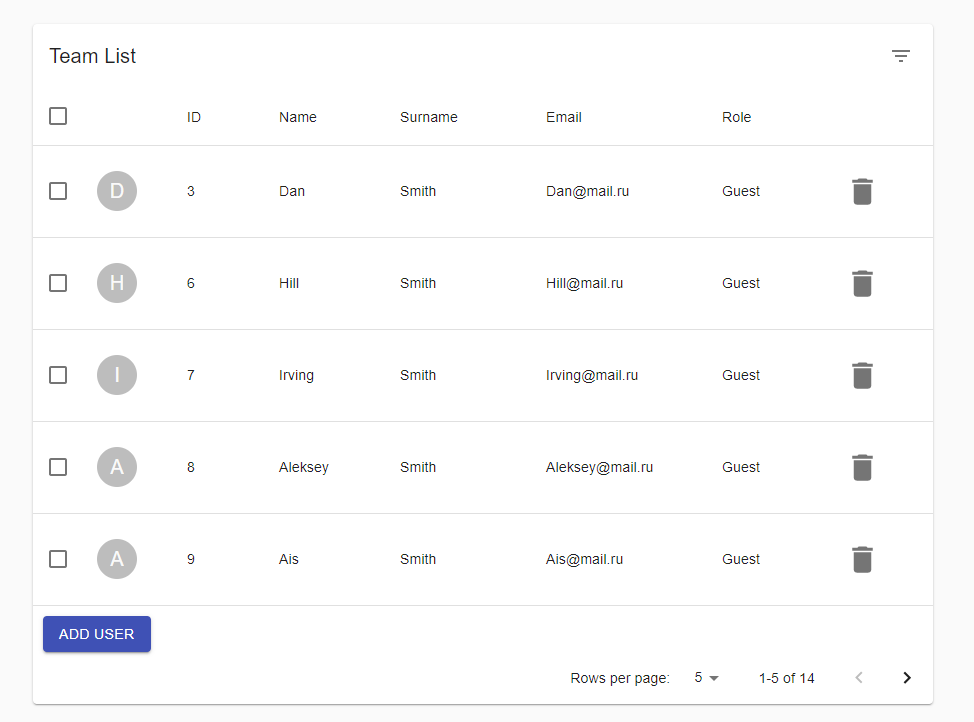


Рисунок 23 – Список всех сотрундников

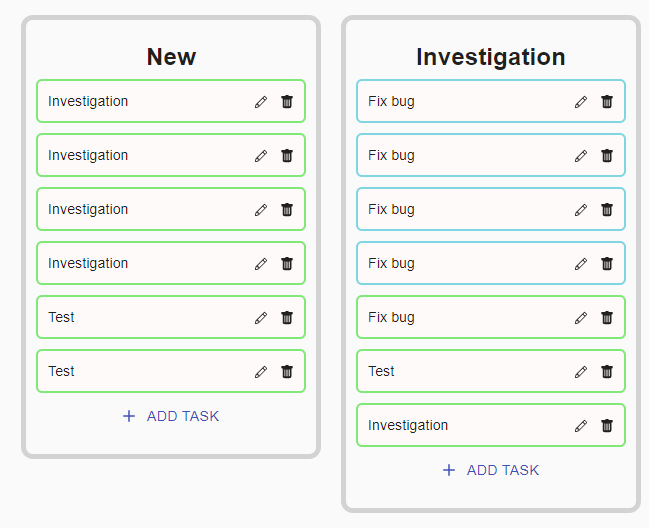
Задачи можно перетаскивать в различные состояния. Например перетащим первую задачу из первого состояния в другое: видно, что задача приобрела новое состояние. 

Рисунок 24 – Новое состояние задачи “Investigation”

Так же каждый тип задачи имеет свой определённый цвет. При необходимости можно добавить новую задачу нажав на кнопку Add Task. Так же задачи можно удалять(иконка мусорного ведра) и редактировать (иконка карандаша).

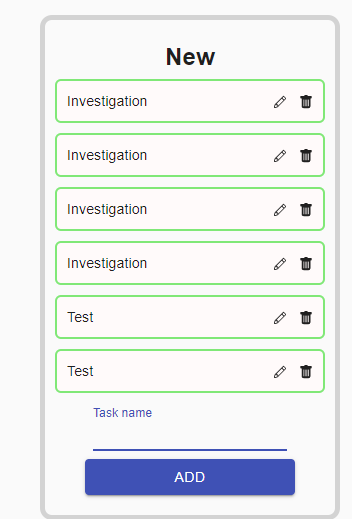


Рисунок 25 – Добавление новой задачи

Задачи можно осмотреть подробнее при нажатии на иконку карандаша, после этого появится всплывающее меню с описанием задачи. Там будут представлены описание задачи, её имя, владелец, комментарии, тип и её статус. Все эти данные могут быть изменены.

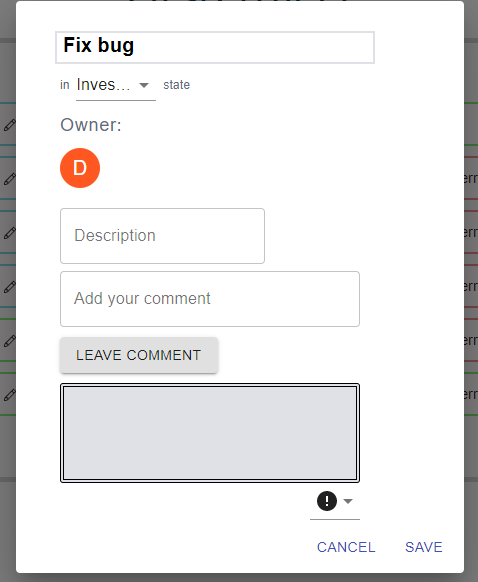


Рисунок 26 – Подробное описание задачи

Для того, чтобы пригласить пользователя в качестве коллаборатора, нужно нажать на соответствующую кнопку на таблице с пользователями. Появится специальная формочка, туда нужно ввести электронную почту пользователя.

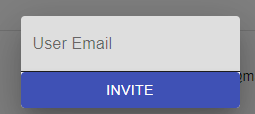


Рисунок 27 – Форма приглашения пользователя

Заключение

В ходе представленной работы был представлен программный модуль, необходимый для оптимизации и ускорения процесса разработки некоторого продукта. Написанное приложение имеет некоторые преимущества, по сравнению с похожими инструментами, например: простота в освоение, подходит для небольших команд, легковестность, возможность удалённого доступа.

Список использованных источников

* 1. Маркин, А. В. Построение запросов и программирование на SQL. Учебное пособие / А.В. Маркин. - М.: Диалог-Мифи, 2014. - 384 c.
  2. [Бхамидипати](http://oz.by/books/more106510.html?id_search=127378#writer), К. SQL. Справочник программиста/ К.Бхамидипати–М.: Эком, 2003. – 304 с.
  3. Бьюли, А. Изучаем SQL / А.Бьюли, Э. Оппель. – М.: Символ, 2007. – 312 с.
  4. Жилинский, А. Самоучитель Microsoft SQL Server 2005 / А.Жилинский – СПб: BHV, 2004. – 224 с.
  5. Форта, Бен Освой самостоятельно SQL за 10 минут / Бен Форта. - М.: Вильямс, 2015. - 288 c.
  6. Павловская, Т. А. С#. Программирование на языке высокого уровня: учебник для вузов / Т. А. Павловская. – СПб. : Питер, 2007. – 432 с.: ил.
  7. Подбельский, В. В. Язык С#. Решение задач / В. В. Подбельский. – М.: Инфра-М, 2014. – 296 с.
  8. Троелсен, Э. Язык программирования С# 4.0 и платформа .NET 4.0 : пер. с англ. / Э. Троелсен. – М. : Вильямс, 2011. – 1392 с.
  9. Дейт, К. Дж. SQL и реляционная теория. Как грамотно писать код на SQL / К.Дж. Дейт. - М.: Символ-плюс, **2017**. - 480 c.
  10. Еремин И.И., Астафьев Н.Н. Введение в теорию линейного и выпуклого программирования -М.: Наука, 1976 г. – 239 с.
  11. Общий курс высшей математики для экономистов. Учебник / под ред В.И. Ермакова.- М.: ИНФА - М. - 656 с. - (серия «высшее образование»).
  12. Грабер, Мартин SQL для простых смертных / Мартин Грабер. - М.: ЛОРИ, 2014. - 378 c.