МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПЕУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 05 01 Информационные системы и технологии

Направление специальности 1-40 01 02 03 Информационные системы и технологии (издательско-полиграфический комплекс)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированное программирование»

Тема «Программное средство самоорганизации по методу agile Катерины Ленгольд»

Исполнитель

Студент 2 курса группы 2 Антонович Юлия Олеговна

(Ф.И.О.)

Руководитель работы ассистент Чевжик Е. А. (учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Пацей Н.В.

(подпись)

Минск 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПЕУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий  
Кафедра информационных систем и технологий

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_ Н.В Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-40 01 05 03 Информационные системы и технологии  Студент: Антонович Ю.О. | Группа: 2 |
| **Тема:** Программное средство самоорганизации по методу agile Катерины Ленгольд | |

**1. Срок сдачи студентом законченной работы**: "май 2019 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС должно:

* Помогать спланировать спринт на основе шаблона;
* Выводить на печать результат планирования спринта;
* Давать возможность сделать список задач и отметить выполненные;
* Давать возможность составлять списки и выводить их на печать.

**2.2.** Программное средство должно быть выполнено с использованием ООП, взаимодействовать с базой данных, реализовано под ОС Windows. Отображение, бизнес-логика и хранилище данных должны быть максимально независимы друг от друга для возможности расширения. В разработке использовать язык C#. Управление программой должно быть интуитивно понятным и удобным. Использовать архитектурные шаблоны и шаблоны проектирования.

**3. Примерное содержание расчетно-пояснительной записки**

(перечень вопросов, подлежащих разработке)

* Введение;
* Обзор литературы и аналогов (алгоритмы решения, обзор литературы);
* Постановка задачи;
* Проектирование программного средства;
* Руководство пользователя;
* Тестирование программного средства;
* Заключение;
* Список используемых источников;
* Приложения.

**4. Форма представления выполненной курсовой работы:**

* Теоретическая часть курсового проекта должны быть представлены в формате MS Word. Оформление записки должно быть согласно правилам.
* Необходимые схемы, диаграммы и рисунки допускается делать в MS Office Visio, VS или других редакторах.
* Листинги программы представляются частично в приложении.
* К записке необходимо приложить CD (DVD).

**Календарный план**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 21.02.2019 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 7.03.2019 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 21.03.2019 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 04.04.2019 |  |
| 5 | Кодирование программного средства | 18.04.2019 |  |
| 6 | Тестирование и отладка программного средства | 02.05.2019 |  |
| 7 | Оформление пояснительной записки | 07.05.2019 |  |
| 8 | Сдача проекта | 20.05.2019 |  |

**5. Дата выдачи задания** 21.02.2019

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Чевжик Е.А.*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Антонович Ю.О.*

(дата и подпись студента)

Оглавление

[ВВЕДЕНИЕ 5](#_Toc10008377)

[1. Аналитический обзор литературы по теме проекта 7](#_Toc10008378)

[2.Анализ требований к программному средству и разработка функциональных требований 9](#_Toc10008379)

[3.Проектирование программного средства 11](#_Toc10008380)

[3.1Описание модели 11](#_Toc10008381)

[3.2Графическое представление модели 11](#_Toc10008382)

[3.3 Обобщенная структура 12](#_Toc10008383)

[3.4 Диаграммы UML, взаимосвязь всех компонентов 14](#_Toc10008384)

[4.Тестирование, проверка работоспособности и анализ полученных результатов 16](#_Toc10008385)

[4.1 Тестирование 16](#_Toc10008386)

[5.Руководство по использованию программного средства 18](#_Toc10008387)

[ЗАКЛЮЧЕНИЕ 22](#_Toc10008388)

[СПИСОК ЛИТЕРАТУРЫ 23](#_Toc10008389)

[Приложение 1 24](#_Toc10008390)

[Листинг 26](#_Toc10008391)

## **ВВЕДЕНИЕ**

Навык планирования сегодня является краеугольным камнем в становлении успешной и разносторонне развитой личности. Грамотно спланированный день будет проведен значительно более эффективно чем день, прошедший в суматохе без плана. Кроме того, многие техники планирования подразумевают подведение итогов дня, недели, года. Таким образом, планирование помогает не только в регулировании собственной жизни, но и в самоанализе.

Человек, анализирующий свои поступки, поведение, окружение, успехи имеет большую склонность к объективному восприятию себя и среды, в которой он находится. Уделяя достаточное время самоанализу, человек имеет большую склонность к улавливанию тонких жизненных моментов, что, в свою очередь, помогает эффективно реагировать на любые события.

Также такой инструмент планирования как подведение итогов дня помогает снять когнитивную нагрузку и подготовиться ко сну. А вечернее планирование следующего дня помогает минимизировать возникновение стрессовых ситуаций. Таким образом планирование помогает не только в личностном развитии, но и в сохранении здоровья.

Среди причин, по которым люди отказываются от планирования, называют следующие:

* Вдохновленность красивыми фотографиями в блогах и неумение повторить красивый почерк и оформление.
* Привычка вести все записи электронно и убежденность авторов методик в необходимости вести записи на бумаге сталкиваются, убивая желание начинать.
* Желание работать с чем-то более красивым чем exel таблицы и нежелание получать более сложный инструмент.

Цель разрабатываемого мной программного средства – создание инструмента для планирования, основанного на техниках, представленных в книге «Космос. Agile-ежедневник для личного развития» Катерины Ленгольд.

Вопреки большому количеству представленных в книгах методик планирования, для разработки программного средства будут выбраны только ключевые из них, поскольку разрабатываемое программное средство должно быть максимально просто в понимании и не заставлять пользователя лишний раз задумываться над требуемыми от него действиями.

Также важно предоставить удобный и понятный инструмент выставления приоритета каждой вводимой задачи. В разрабатываемом мной программном средстве это реализовано графически.

Сама тема разработки инструмента для планирования не нова, однако, как показывает практика, выбор подходящего для планирования средства – вопрос крайне индивидуальный, поэтому важно разработать программное средство, не имеющее абсолютного аналога, однако с учетом плюсов и минусов других средств планирования.

Аудитория данного программного средства представлена людьми, которые только начинают свой путь в планировании и не хотят углубляться во времязатратные самоанализы.

Важно учитывать, что большинство пользователей не прочтёт пользовательскую инструкцию, поэтому рекомендации и подсказки должны быть в интерфейсе программного средства.

Поскольку планирование – достаточно личный процесс, важно организовать авторизацию в приложении с помощью пароля. Таким образом пользователь получит возможность видеть свои личные данные только когда он сам этого захочет.

Совокупность всех приведенных выше положений приведет к созданию качественного и полезного продукта, который будет отвечать требованиям безопасности.

Цели проекта:

* Совершенствование умений написания программного кода;
* Совершенствование умений настройки SQL сервера;
* Формирование навыков проектирования баз данных;
* Формирование навыков проектирования и разработки пользовательского интерфейса;
* Научиться проводить тестирование продукта;
* Формирование умения строить архитектуру проекта, оставляя для него возможность масштабирования.

Задача проекта:

Разработать архитектуру приложения с учетом интересов потенциального потребителя и создать программное средство с user-friendly интерфейсом и максимально сегментированной базой данных. Использовать в разработке объектно-ориентированный язык программирования. Также настроить SQL сервер и создать базу данных, обеспечив простой доступ к ней из приложения.

# **1. Аналитический обзор литературы по теме проекта**

В основу реализации разрабатываемого программного продукта ложатся две книги: «Джедайские техники» Макса Дорофеева и «Космос. Agile-ежедневник для личного развития» Катерины Ленгольд. Однако ни один из подходов не будет реализован в полной мере для достижения целей, описанных во введении. Таким образом разрабатываемое программное средство станет более универсальным аналогом печатным изданиям.

Рассмотрим основные идеи, предлагаемые авторами вышеназванных изданий.

Теория Дорофеева строится вокруг экономии мыслетоплива. В контексте Джедайских техник мыслетопливо – ресурс, который нужен для эффективного мышления и самоконтроля. А усталость в рабочем процессе, в свою очередь, является результатом расхода мыслетоплива.

Автор убежден, что при рациональном использовании мыслетоплива можно в разы и даже десятки раз повысить свою продуктивность.

Согласно Дорофееву, есть три основных момента, помогающих нам экономить мыслетопливо:

* Разгрузка памяти. Для этого автор рекомендует иметь единое место (условно называемое папкой Входящие) для записи текущих планов, целей и задач. Также автор рекомендует для мысленной разгрузки использовать список спусковых крючков (приложение 1).
* Регулярная плановая очистка Входящих. Список Входящих должен обрабатываться, но не выполняться. Все записи должны распределяться по новым условным папкам. Первая папка – Задачи – четко, конкретно и понятно сформулированные действия, выполнение которых не требует осмысления или другой сложной когнитивной нагрузки. Вторая папка – Проекты или дела. В неё должны попадать большие куски активностей, выполнение которых требует когнитивной нагрузки, осмысления порядка действий, формирования задач. Третья папка – Информация – это данные, которые напрямую не призывают к выполнению действий, однако могут понадобиться в процессе выполнения задач.
* Как эффективный способ работы называется работа с Pomodoro техникой.

В реализуемом программном средстве будет предложен механизм написания списков и листа задач, с которыми можно работать в том числе по методике Джедайских техник, однако приложение не ставит своей целью залезть в мысли пользователя и понять, как он думает (в рамках методики или нет). Важно чтобы разрабатываемый инструмент был подходящим для людей, прочитавших Дорофеева.

Обращаясь к Катерине Ленгольд отметим, что суть ее техники – правильное планирование личного времени и целей, для более эффективного получения конечного результата.

Из Ленгольд будет взят принцип работы спринтами (9-недельными периодами) над формированием привычек. С начала нового спринта будет запрос на формулирование трех ключевых привычек, над формированием которых будет производиться работа в ходе спринта. К каждой привычке нужно написать обоснование, почему пользователю формирование этой привычки необходимо.

Идея спринтов лежит в основе методологии гибкой разработки Scrum, позволяющей компаниям разрабатывать программное обеспечение быстро и регулярно.

Личное планирование мало чем отличается от разработки ПО, и итеративный подход со спринтами оказывается весьма кстати. Вместо того, чтобы ставить перед собой огромные задачи на год вперед, вы фокусируетесь на конкретных, относительно небольших задачах. В конце спринта - оцениваете результаты, делаете работу над ошибками и выбираете вектор движения на следующие 9 недель.

Несмотря на важность правильного целеполагания, автор этого планнера уверена, что ничто так не влияет на наш успех в жизни, как привычки. Ежедневное повторение одного и того же действия перестраивает нейронные связи в нашем мозге, и с каждым разом нам требуется все меньше силы воли, так как процесс становится автоматическим.

На каждый спринт вы будете выбирать новые привычки для внедрения - и с помощью планнера их закреплять навсегда, делая автоматическими.

Кроме того, разработчиком предлагается вариант использования agile-подхода не только с целью саморазвития, но и с целью планирования. Таким образом можно ставить цель, разбивать ее на более мелкие (этапы), формулировать актуальность. Так мы приходим к подходу, описанному в джедайских техниках, не меняя функционала.

Аналоги данного продукта могут быть печатные и электронные версии книг, названных выше, однако программного средства для работы по описанным в книгах техникам сегодня не существует. Разрабатываемое программное средство будет представлять собой реализацию двух техник сразу, но не в полном, в облегченном виде, доступном пользователю, только начинающему знакомство с планированием.

# **2.Анализ требований к программному средству и разработка функциональных требований**

Разрабатываемое приложение должно содержать форму авторизации. Уровень доступа будет один – для пользователя. Форма реализации продукта – десктоп приложение.

Десктоп приложение должно быть реализовано с использованием языка С#. Технология для реализации – WPF. База данных приложения должна взаимодействовать с Microsoft SQL server.

Для вывода на печать использовать System.Windows.Controls.PrintDialog.

В приложении требуется сделать три основные вкладки меню – Планирование, Саморазвитие, Списки.

В разделе Планирование реализовать возможность дописать новый план, отобразить время написания плана (автоматический вывод системного времени), отметить выполненные дела и удалить их по нажатию на соответствующую кнопку.

В разделе Саморазвитие требуется реализовать своеобразный макет для работы с agile системой Катерины Ленгольд. Сделать содержание приближенным к оригиналу (Рисунок 2.1). Дать возможность вывода на печать.



Рисунок 2.1 – страница agile ежедневника Катерины Ленгольд для реализации в приложении

В разделе Списки сделать возможность добавления нового элемента в список и удаления списка. Реализовать вывод на печать содержимого списка.

Для обучения пользователя в процессе использования приложения сделать вкладку меню Помощь, где интерактивно кратко дать описания каждому разделу приложения.

Приложение реализовать в цветовой гамме, представленной на рисунке 2.2.



Рисунок 2.2 – цветовая гамма для приложения

Сделать приложение динамическим. Согласно исследованиям, современному человеку сложно долго фокусироваться на статических вещах, поэтому если приложение будет статическим, пользователь с большой долей вероятности будет несознательно реагировать на внешние раздражители. Этот феномен получил название клипового мышления, а разрабатываемое приложение должно получить динамическое меню и анимированные элементы управления.

# **3.Проектирование программного средства**

## **3.1Описание модели**

Для программного средства в курсовом проекте был выбран один из самых распространенных способов разработки программных модулей – клиент-серверная модель. Приложение разбивается на 2 части: клиентская часть приложения – это графический интерфейс. Графический интерфейс отображается в самом приложении. Вторая часть – серверная часть приложения – SQL сервер, обрабатывающий запросы пользователя.

## **3.2Графическое представление модели**

Для реализации поставленной задачи была создана база данных. Для её создания использовался Microsoft SQL Server.

База данных – совокупность данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранение и манипулирования данными, независимая от прикладных программ.

Система управления базами данных – совокупность программ и языковых средств, предназначенных для управления данными в базе данных, введения базы данных и обеспечения взаимодействия ее с прикладными программами.

База данных программного средства состоит из 4 таблиц, и ее схема изображена на рисунке 3.2.1

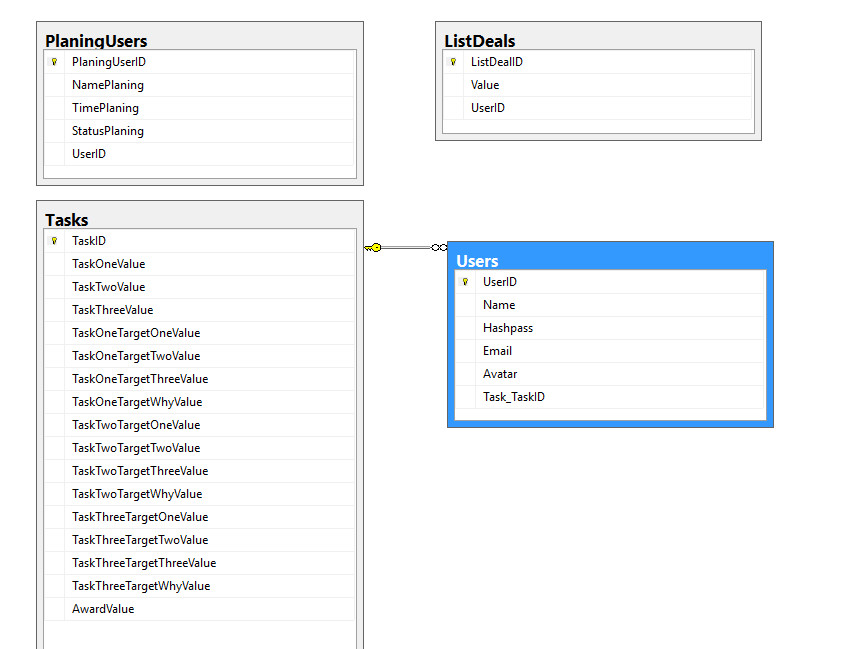


Рисунок 3.2.1 -Схема базы данных

Для хранения данных о пользователе предназначена таблица Users с полями:

* UserID
* Name
* Hashpass
* Email
* Avatar
* Task\_TaskID

Для хранения данных о планах предназначена таблица PlaningUsers с полями:

* PlanningUserID
* NamePlanning
* TimePlanning
* StatusPlanning
* UserID

Для хранения результатов scram-планирования пользователя предназначена таблица Tasks с полями:

* TaskID
* TaskOneValue
* TaskTwoValue
* TaskThreeValue
* TaskOneTargetOneValue
* TaskOneTargetTwoValue
* TaskOneTargetThreeValue
* TaskTwoTargetOneValue
* TaskTwoTargetTwoValue
* TaskTwoTargetThreeValue
* TaskThreeTargetOneValue
* TaskThreeTargetTwoValue
* TaskThreeTargetThreeValue
* Award Value

## **3.3 Обобщенная структура**

Решение Restorator представлено одним проектом Restorator, имеющим структуру, представленную на рисунке 3.3.1.

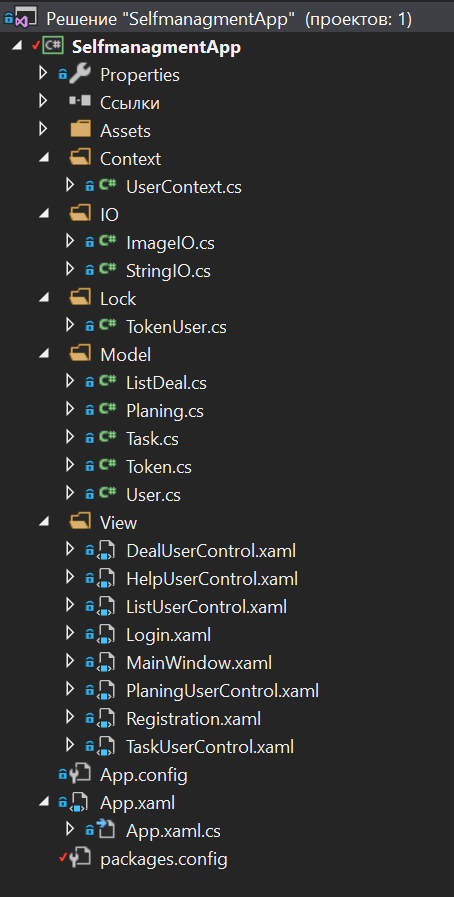


Рисунок 3.3.1 – Структура проекта

|  |  |
| --- | --- |
| Properties | Свойства проекта, содержит информацию о сборке, используемых ресурсах и настройках |
| Ссылки | Перечень сборок, используемых в проекте |
| Properties | Папка со свойствами проекта |
| App.config | Файл с параметрами проекта |
| App.xaml | Класс Application |
| Assets | Папка с картинками, используемыми в интерфейсе |
| UserContext.cs | Контекст для работы с БД |
| ImageIO.cs | Вспомогательный класс для работы с изображениями |
| StringIO.cs | Вспомогательный класс для работы со строками |
| TokenUser.cs | Токин пользователя для логина |
| ListDeal.cs | Макет данных пользовательского списка для записи в БД |
| Planing.cs | Макет данных планов для записи в БД |
| Task.cs | Макет данных поставленных задач для записи в БД |
| Token.cs | Макет данных токена для записи в БД |
| User.cs | Макет данных пользователя для записи в БД |
| DealUserControl.xaml | Окно списка дел |
| HelpUserControl.xaml | Окно Помощь |
| ListUserControl.xaml | Окно списков |
| PlaningUserControl.xaml | Окно планирования |
| TaskUserControl.xaml | Окно постановки задач |
| Login.xaml | Окно логина |
| MainWindow.xaml | Главное окно |
| Registration.xaml | Окно регистрации |

## **3.4 Диаграммы UML, взаимосвязь всех компонентов**

Диаграмма UML – это графическое представление набора элементов, изображаемое чаще всего в виде графа с вершинами (сущностями) и ребрами (отношениями). Диаграммы рисуют для визуализации. Основная цель диаграмм – визуализация разрабатываемой системы с разных точек зрения.

На диаграмме 3.4.1 представлен процесс работы с приложением со стороны пользователя.

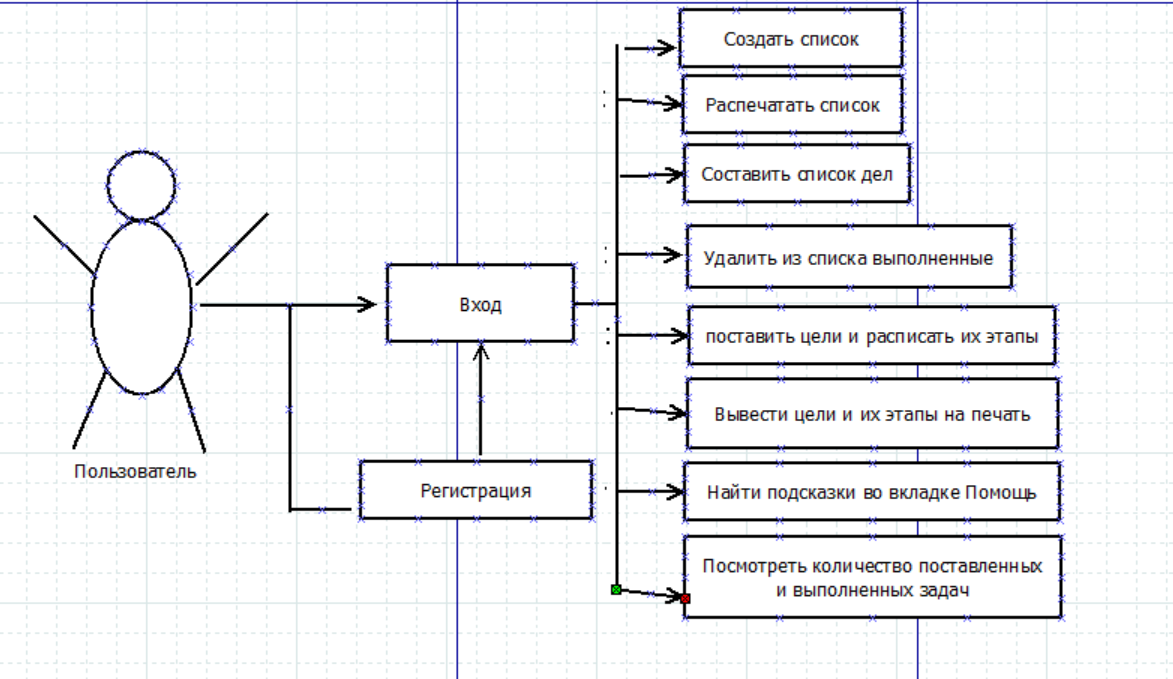


Рисунок 3.4.1 – Принцип работы пользователя

Диаграмма классов проекта представлена на рисунке 3.4.3.

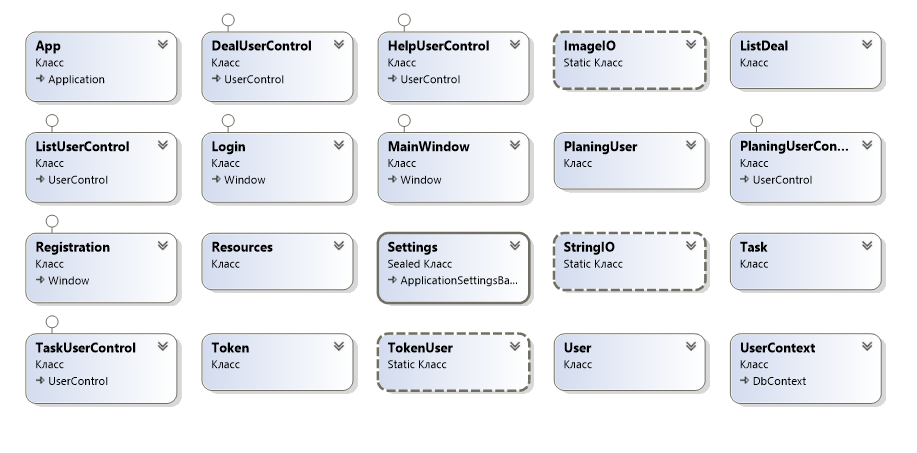


Рисунок 3.4.3 – диаграмма классов

# **4.Тестирование, проверка работоспособности и анализ полученных результатов**

## **4.1 Тестирование**

Для оценки работоспособности разработанного приложения были проведены тесты использование памяти и использование ЦП.

Тестирование использование памяти проводится с целью обнаружить проблемы как утечка памяти.

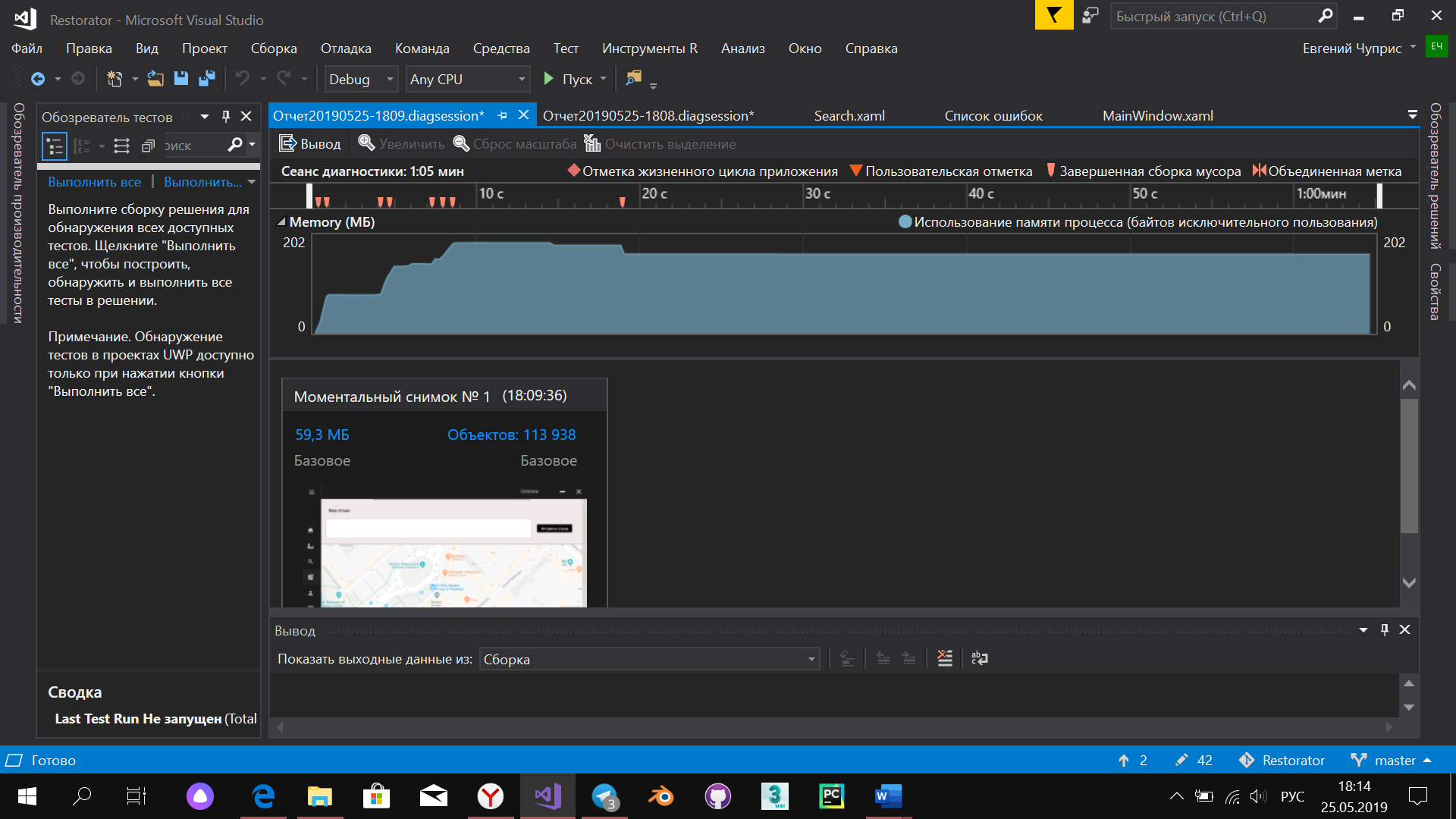


Рисунок 4.1 – Тестирование памяти

Как показано на Рисунке 4.1 тестирование приложения на проблему утечку памяти пройдено успешно. Приложения в процессе своей работы использует 150-190 мб памяти.

Тестирование использование ЦП проводится с целью нахождения степени загруженности центрального процессора при работе программы.

Рисунок 4.2 – Тестирование памяти

Как показано на Рисунке 4.2 тестирование приложения на загруженность процессора основная нагрузка на приложение (63%) при запуске приложения. На запуск программы и загрузке страницы о лучшем ресторане тратиться в общем 6 секунд при загрузке в 60%. На загрузку каталога всех заведений используется 14.71% ЦП. Стоит заметить, что нагрузка на процессор не постоянная, а в среднем 0.5-0.9 секунды. Данный результат можно считать хорошим.

На рисунке 4.3 показана временная шкала выполнения процессов в приложении.

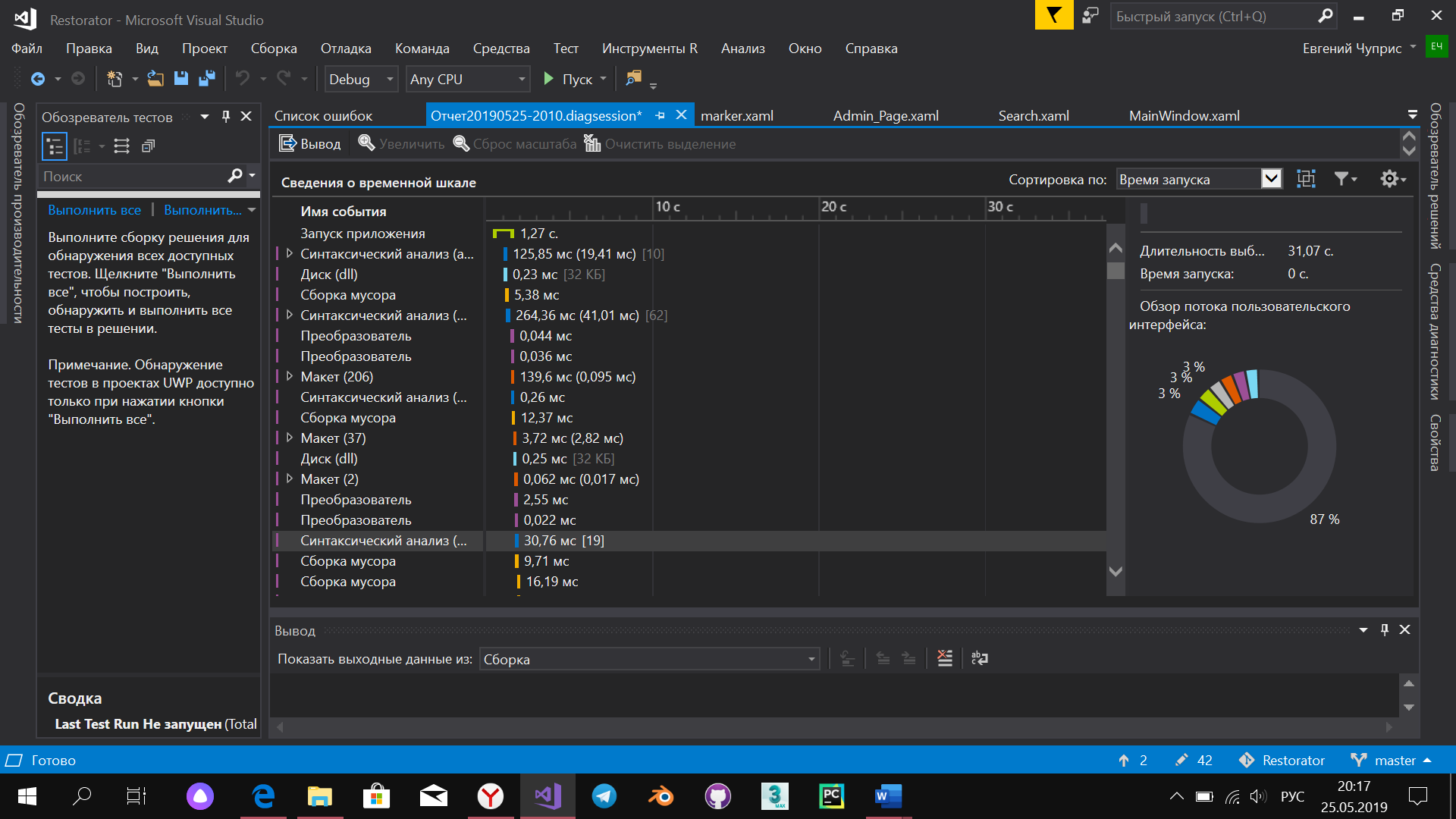


Рисунок 4.2 – Временная шкала

По результату тестирование и получение временной шкалы программного средства можно сделать вывод что из во время работы нашего приложения 87% времени свободно от процессов, что говорит о хорошей оптимизации приложения.

Протестировав приложение можно сделать вывод, что программное средство является нетребовательным к системным характеристикам приложением и может запуститься практически на любом устройстве с оперативной памятью не ниже 512 мб.

# **5.Руководство по использованию программного средства**

При запуске приложения открывается окно входа пользователя (рисунок 5.1) с возможностью перехода к окну регистрации (рисунок 5.2).

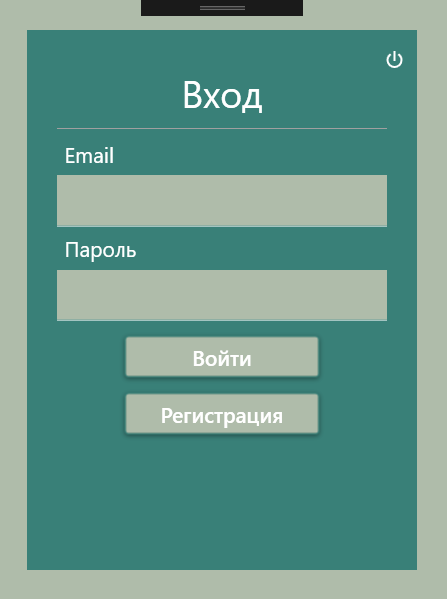


Рисунок 5.1 – вход

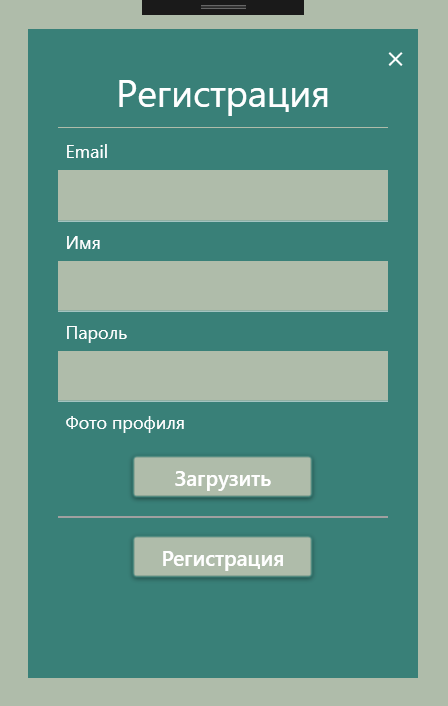


Рисунок 5.2 - регистрация

Имея данные для входа, пользователь войдет в приложение по нажатию на кнопку Войти.

После входа в приложение пользователю откроется вкладка меню Помощь (Рисунок 5.3) с всплывающими подсказками при наведении курсора на картинки.

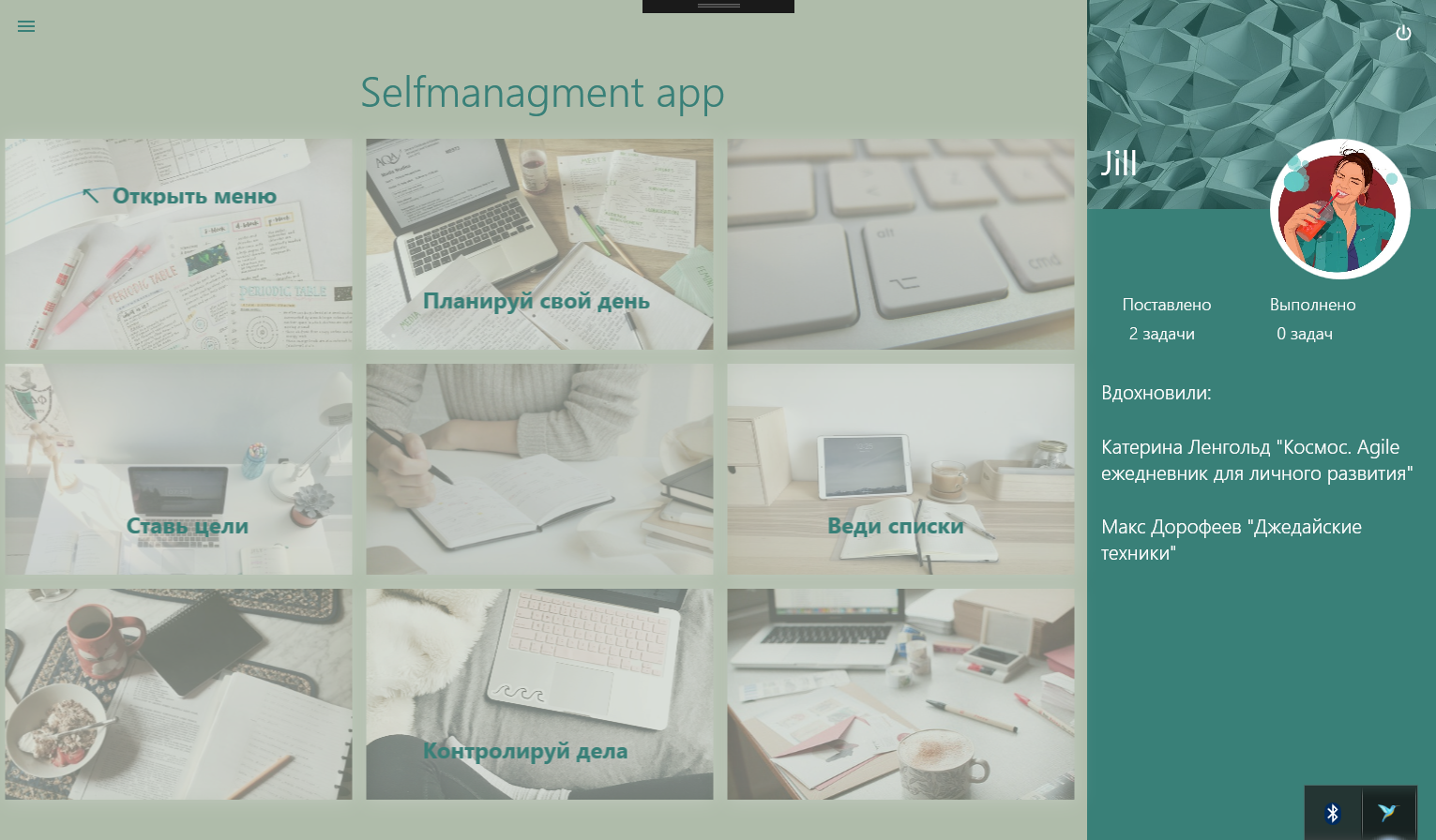


Рисунок 5.3 – вкладка меню Помощь

По нажатию на иконку пользователь может открыть меню для перехода по другим вкладкам.

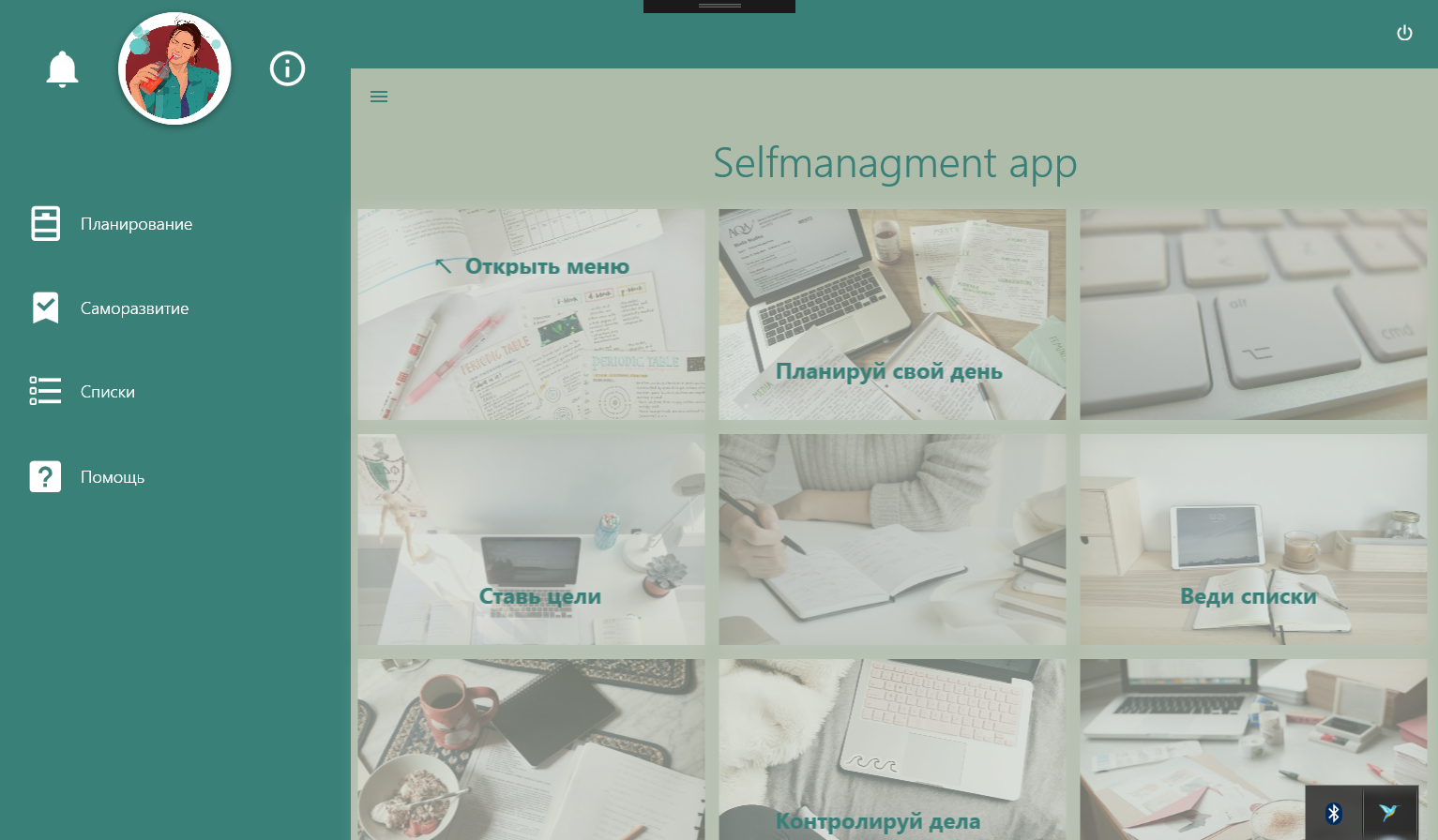


Рисунок 5.3 – меню

При переходе по вкладке меню Планирование произойдет переход ко странице, представленной на рисунке 5.4.

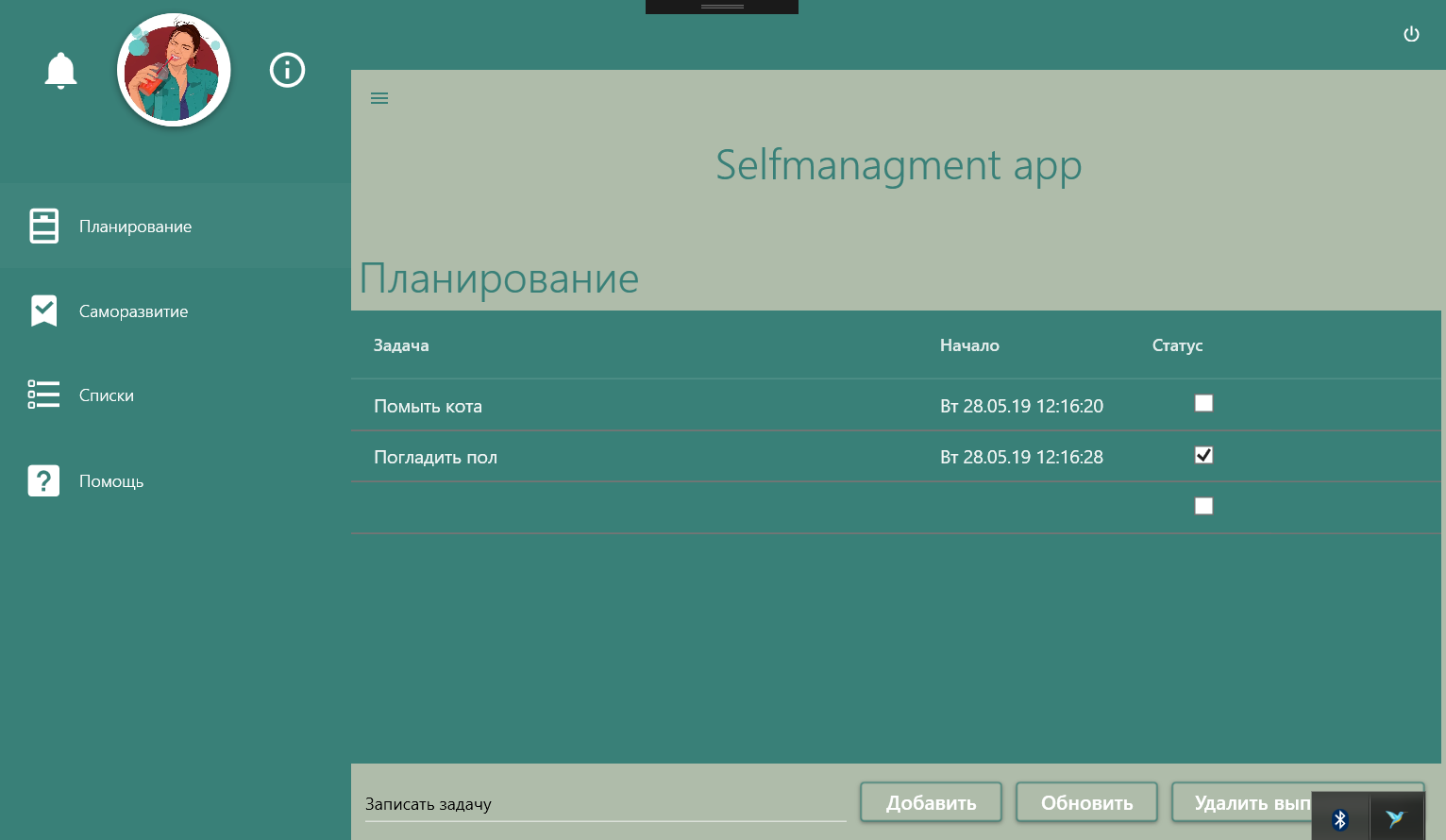


Рисунок 5.4 – вкладка Планирование

Добавление новой задачи происходит после нажатия кнопки Добавить. Чтоб отметить задачу как выполненную нужно поставить галочку и нажать обновить. При нажатии на Удалить выполненное удаляются все задачи, отмеченные галочками.

При переходе по вкладке меню Саморазвитие произойдет переход ко странице, представленной на рисунке 5.5.

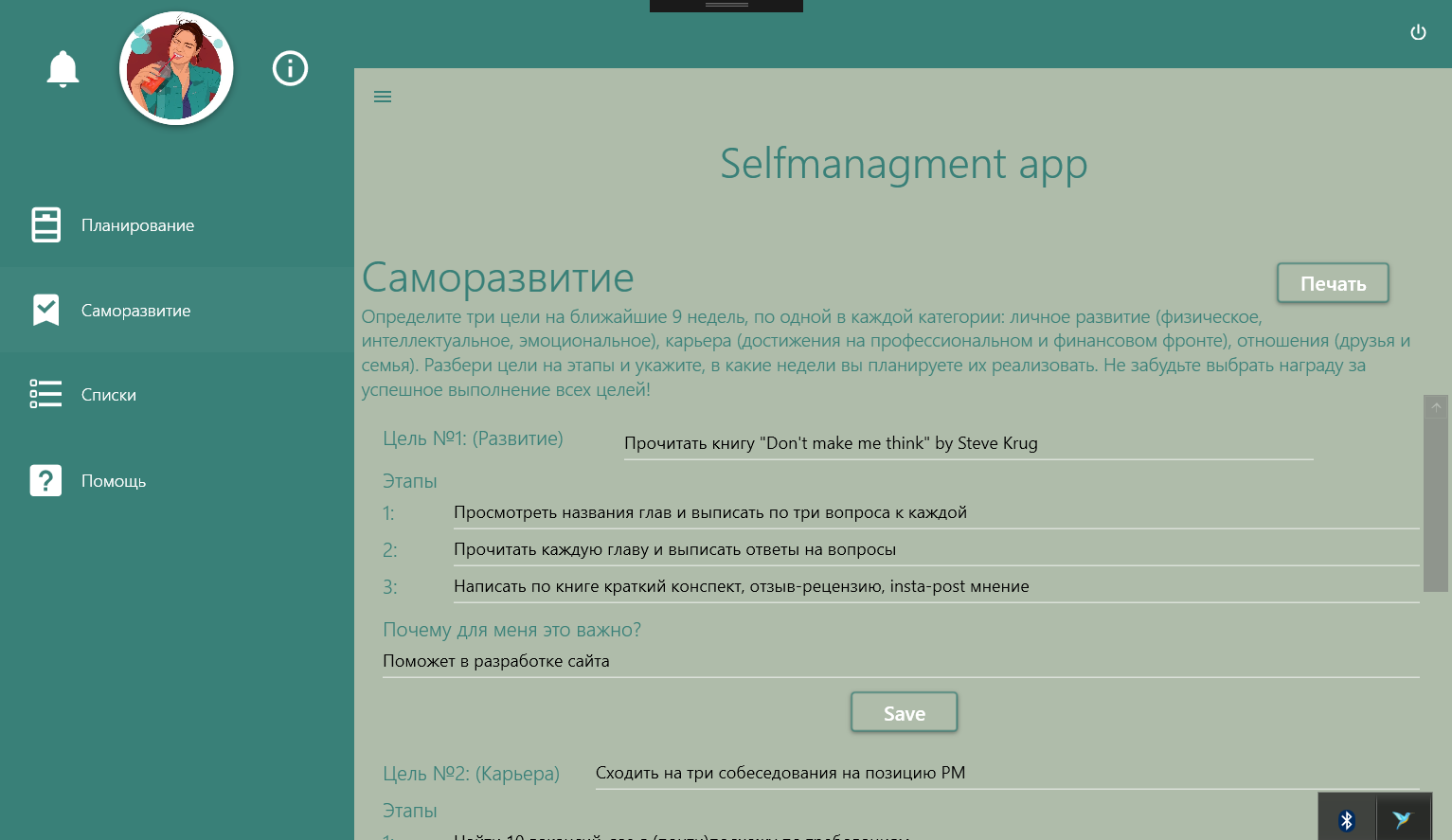


Рисунок 5.5 – вкладка Саморазвитие

Кнопка Save Сохраняет вводимые данные. Печать открывает меню настройки параметров печати (рисунок 5.7).

При переходе по вкладке меню Списки произойдет переход ко странице, представленной на рисунке 5.6.

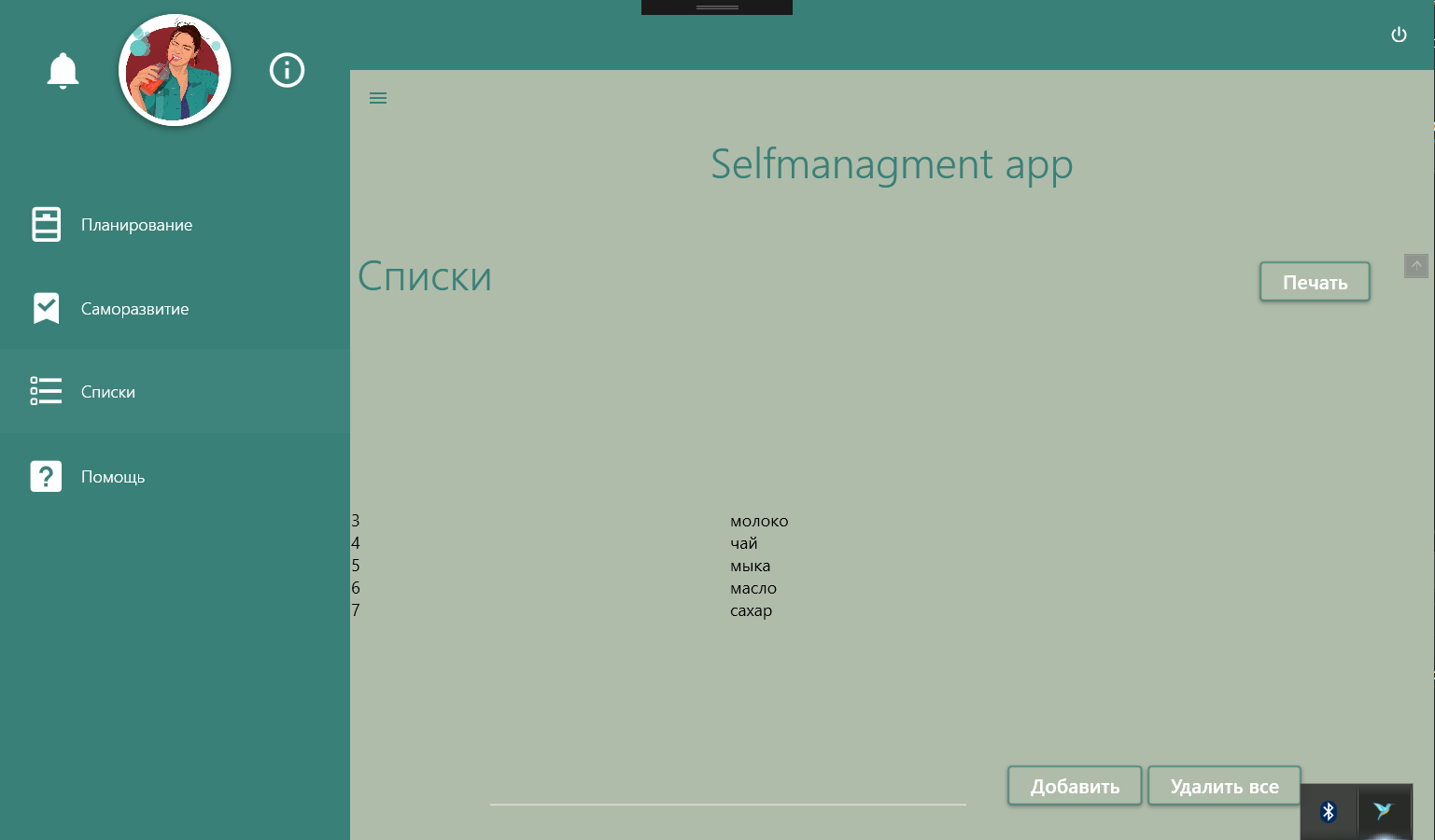


Рисунок 5.6 – вкладка Списки

Печать открывает меню настройки параметров печати (рисунок 5.7). Добавление элемента происходит по нажатию на кнопку Добавить. Удаление – Удалить все.

В любой момент пользования приложением пользователь имеет право свернуть меню и тем самым открыть правое меню как на рисунке 5.3.

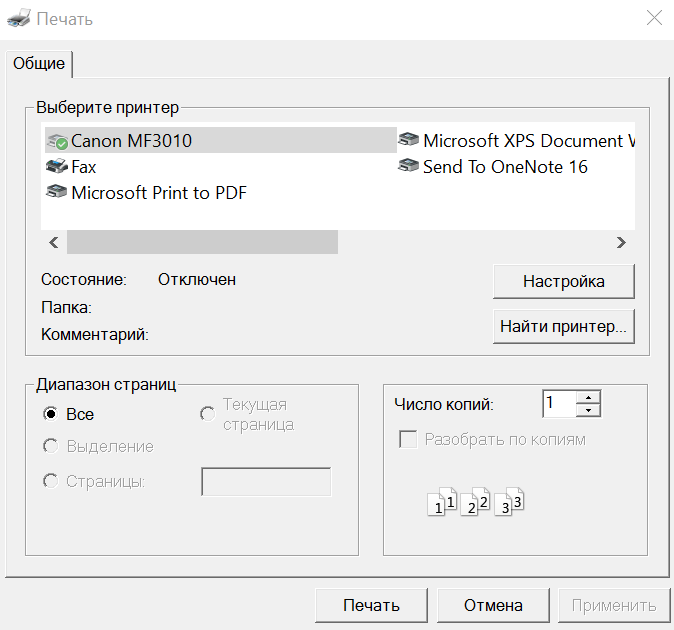


Рисунок 5.7 – диалоговое окно перед выводом информации на печать

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения работы были достигнуты следующие результаты:

* Изучена работа с технологией WPF, практиковалось написание кода на языке C#;
* Изучены остновные концепции книг Катерины Ленгольд и Макса Дорофеева;
* Сформулирована модель данных и приведено ее графическое представление;
* Показана результатами тестирования продукта его работоспособность;
* Разработано руководство по использованию приложения;
* Подготовлена база данных с использованием Entity Framework;
* Испытанно приложение на показатель user-friendly и получены положительные результаты.

# **СПИСОК ЛИТЕРАТУРЫ**

1 MSDN сеть разработчиков в Microsoft [Электронный ресурс] – <http://msdn.microsoft.com/library/rus/>. Дата доступа: 10.05.2017

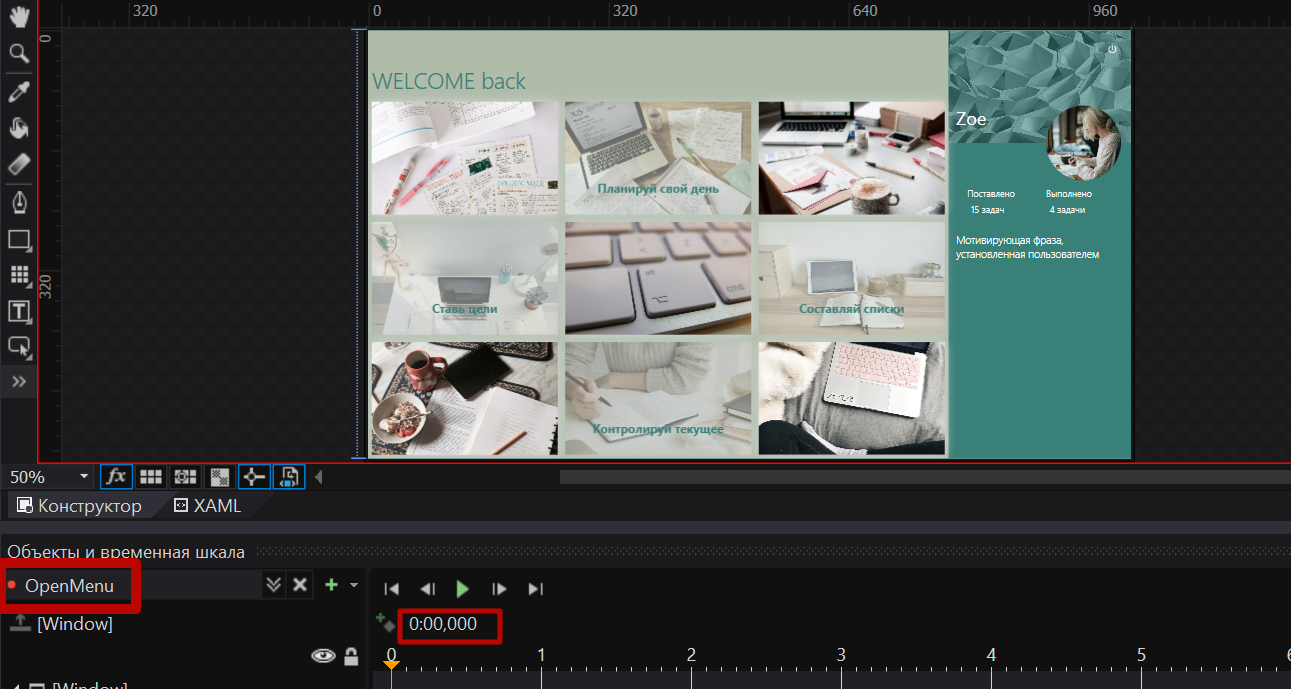
2 Язык C# и .NET Framework [Электронный ресурс] – <http://professorweb.ru/my/csharp/charp_theory/level1/infonet.php>. Дата доступа: 10.05.2017

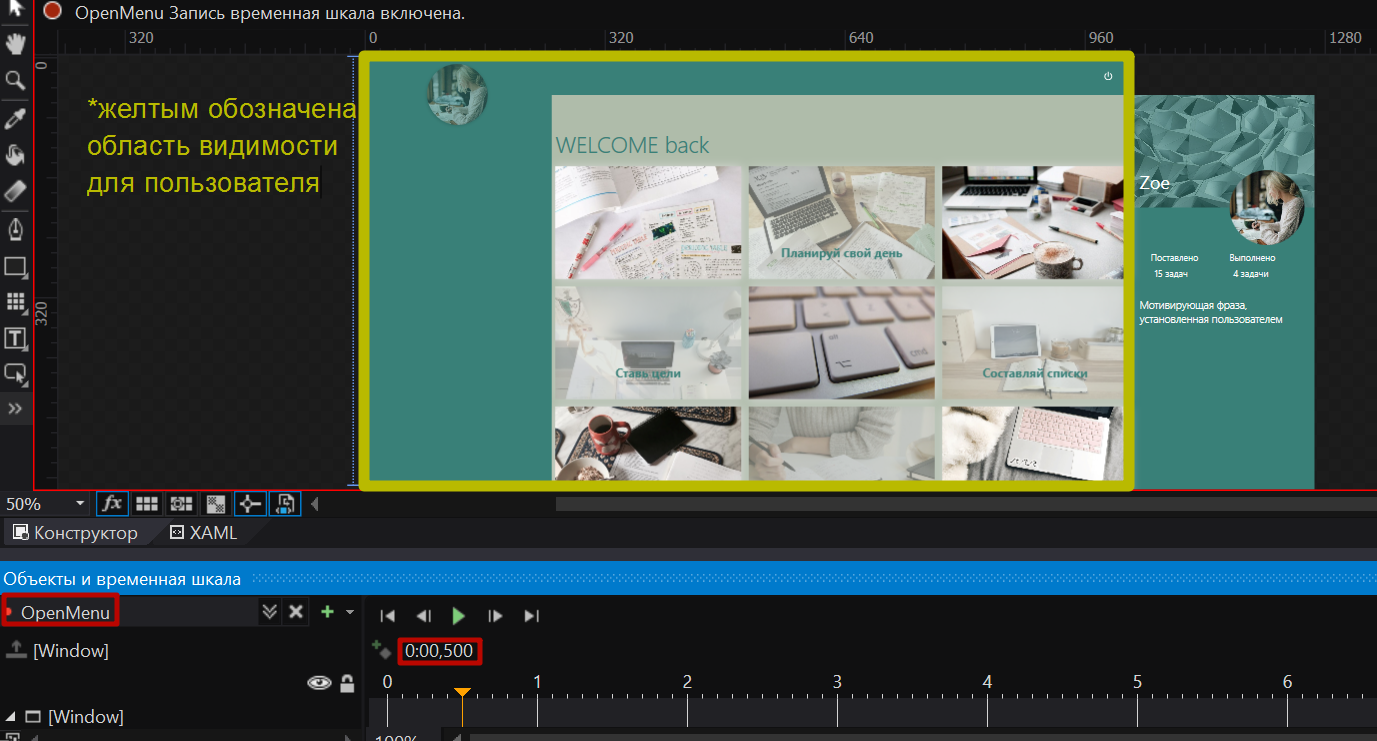
3 Руководство по ADO.NET и работе с базами данных [Электронный ресурс] – <https://metanit.com/sharp/entityframework/1.1.php> Дата доступа: 10.05.2017

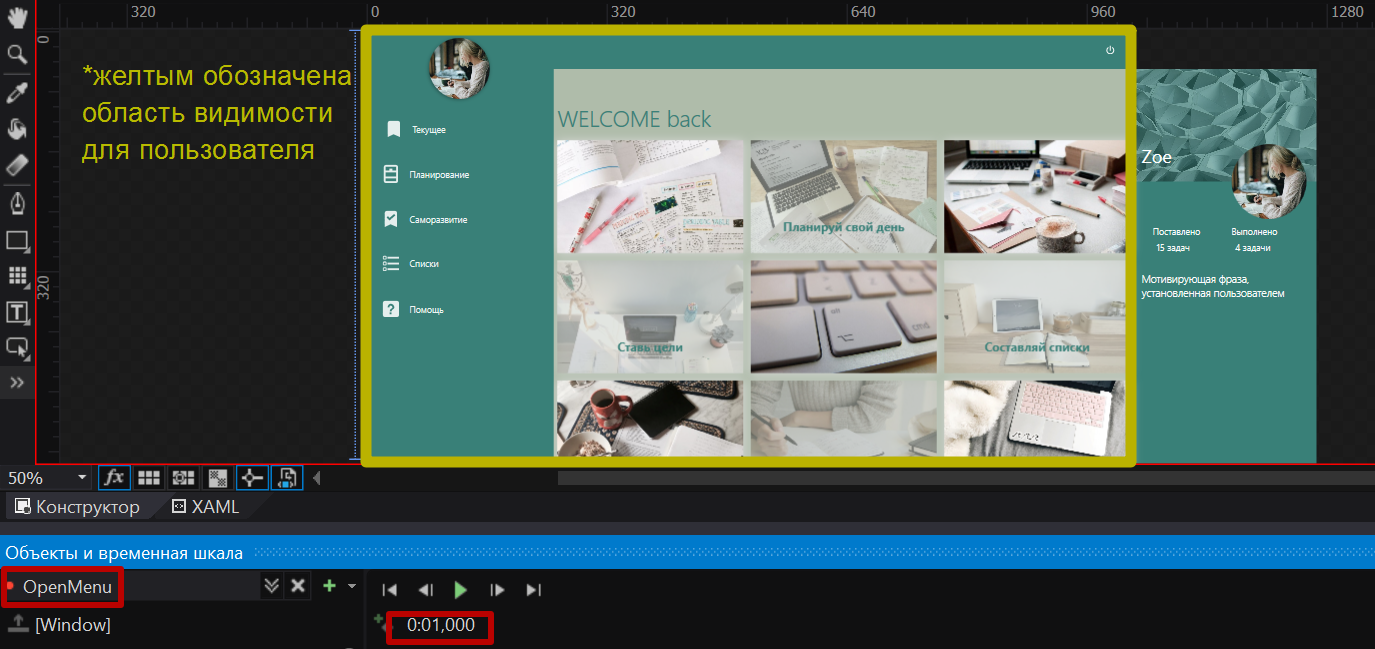
4 Форум для программистов или разработчиков [Электронный ресурс] – <https://stackoverflow.com/>. Дата доступа: 10.05.2017

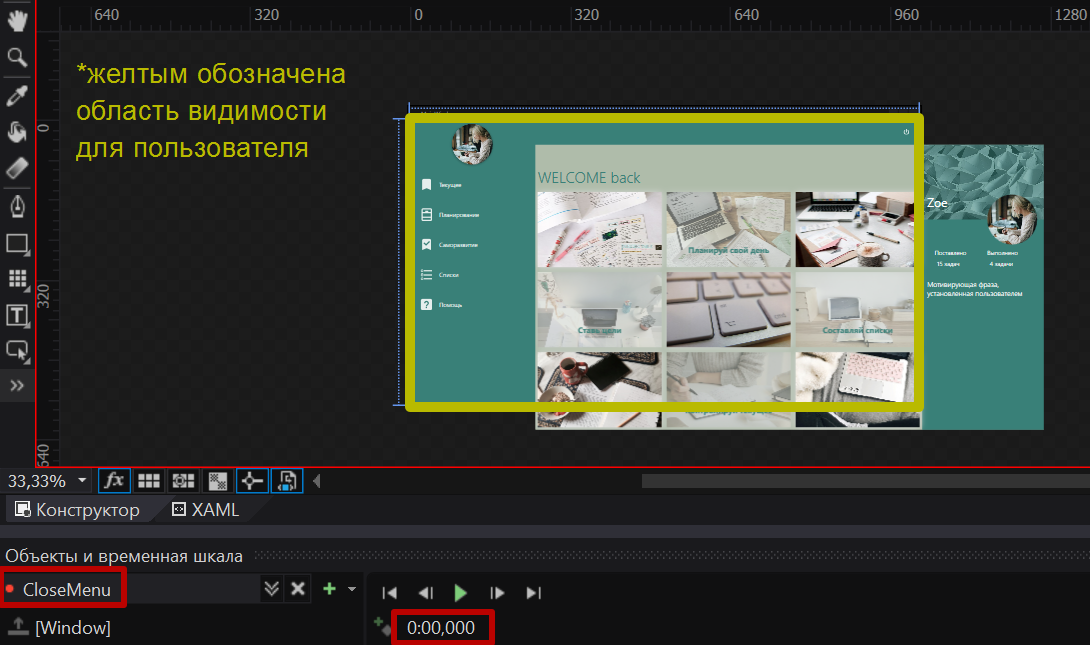
## **Приложение 1**

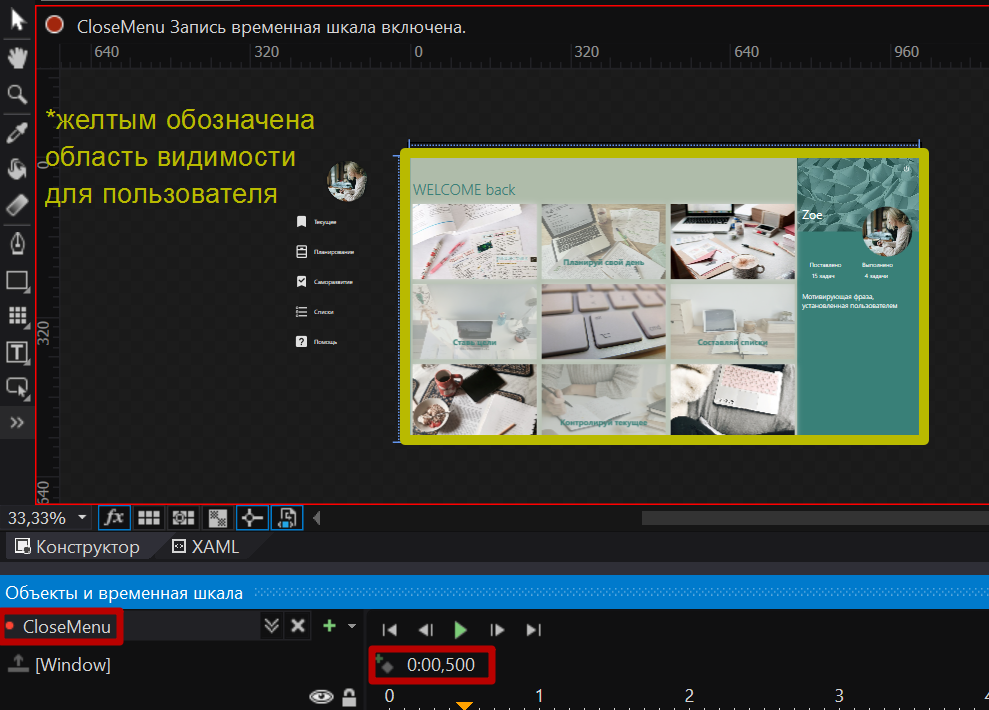
Пошаговый процесс проектирования динамического меню.











## **Листинг**

public static class StringIO

{

public static string GetSHA256(string input)

{

string output = "";

SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes(input)).ToList().ForEach(f => output += f);

return output;

}

}

public static class ImageIO

{

public static byte[] BitmapImageToByte(BitmapImage imageSource)

{

byte[] data;

JpegBitmapEncoder encoder = new JpegBitmapEncoder();

if (imageSource == null)

{

return null;

}

encoder.Frames.Add(BitmapFrame.Create(imageSource));

using (MemoryStream ms = new MemoryStream())

{

encoder.Save(ms);

data = ms.ToArray();

}

return data;

}

public static BitmapImage ByteToBitmapImage(byte[] array)

{

using (var ms = new MemoryStream(array))

{

var image = new BitmapImage();

image.BeginInit();

image.CacheOption = BitmapCacheOption.OnLoad;

image.StreamSource = ms;

image.EndInit();

return image;

}

}

}

public static class TokenUser

{

public static Token Token { get; set; }

public static void CreateToken(User user)

{

if (Token == null)

{

Token = new Token(user);

}

}

}

public class User

{

[Key, DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int UserID { get; set; }

public string Name { get; set; }

public string Hashpass { get; set; }

public string Email { get; set; }

public byte[] Avatar { get; set; }

public Task Task { get; set; }

private User()

{

}

public User(string Name, string Pass, string Email, BitmapImage Avatar)

{

this.Name = Name;

this.Hashpass = StringIO.GetSHA256(Pass);

this.Email = Email;

this.Avatar = ImageIO.BitmapImageToByte(Avatar);

Task = new Task();

}

}