**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  доцент департамента  программной инженерии,  доцент, канд. пед. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С. А. Виденин  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **ПРОГРАММА-АСССИСТЕНТ ДЛЯ ГОСТИНИЧНОГО БИЗНЕСА**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.10.03-01 81 01-1-ЛУ** | | | | | | |
|  | |  | | | | |
| Исполнитель  студент группы БПИ218  \_\_\_\_\_\_\_\_\_ / Жуков Ф.С./  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | | | | |
|  | | | | | | |
|  | | | | |  | |

|  |
| --- |
| УТВЕРЖДЕН  RU.17701729.09.09-01 81 01-1-ЛУ |
|  | | |

**ПРОГРАММА-АССИСТЕНТ**

**ДЛЯ ГОСТИНИЧНОГО БИЗНЕСА**

**Пояснительная записка**

**RU.17701729.10.03-01 81 01-1**

**Листов 36**

# 

# АННОТАЦИЯ

В текущем программном документе представлена пояснительная записка к десктоп приложению «Программа-ассистент для гостиничного бизнеса»

В разделе «Введение» представлены полное и краткое название программы, а также документы, на основании которых ведётся разработка.

В разделе «Назначение разработки» описаны функциональное назначение программы, а также эксплуатационное назначение.

В разделе «Техническое характеристики» описаны задачи, которые стояли на пути разработки данного приложения. Также присутствует описание программы, если быть точнее, описание функционирования основных окон, а также описан формат входных данных, которые должны быть написаны перед запуском программы, и формат выходных данных, которые будут получаться в процессе работы программы.

В разделе «Ожидаемые технико-экономические показатели» представлена ожидаемая потребность, экономические преимущества, а также приведены примеры конкурентных приложений, показаны их сильные и слабые стороны, и приведена сравнительная таблица.

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.101-77[[1]](#vxg2sw6xnnzw): Виды программ и программных документов.
2. ГОСТ 19.102-77[[2]](#mxx84bqj48qg): Стадии разработки.
3. ГОСТ 19.103-77[[3]](#7h7tzitg66cu): Обозначения программ и программных документов.
4. ГОСТ 19.104-78[[4]](#qc70i0s3vb9w): Основные надписи.
5. ГОСТ 19.105-78[[5]](#b4335150hcax): Общие требования к программным документам.
6. ГОСТ 19.106-78[[6]](#j2be0mlzfo): Требования к программным документам, выполненным печатным способом.
7. ГОСТ 19.404–79. Пояснительная записка. Требования к содержанию и оформлению.
8. ГОСТ 19.602-78[[8]](#esu8hyneww59): Правила дублирования, учета и хранения программных документов, выполненных печатным способом.

Изменения к настоящему техническому заданию должны быть оформлены согласно ГОСТ 19.603-78[[9]](#orguoh3mb6zx) и ГОСТ 19.604-78[[10]](#f3q852lse1wh).

СОДЕРЖАНИЕ

[АННОТАЦИЯ 3](#_Toc134637037)

[1. ВВЕДЕНИЕ 5](#_Toc134637038)

[1.1. Наименование программы 5](#_Toc134637039)

[1.2. Краткая характеристика области применения программы 5](#_Toc134637040)

[2. НАЗНАЧЕНИЕ РАЗРАБОТКИ 6](#_Toc134637041)

[2.1 Функциональное назначение 6](#_Toc134637042)

[2.2 Эксплуатационное назначение 6](#_Toc134637043)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc134637044)

[3.1. Постановка задачи на разработку программы 7](#_Toc134637045)

[3.2. Описание алгоритма и функционирования программы 7](#_Toc134637046)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 25](#_Toc134637047)

[4.1. Ориентировочная экономическая эффективность 25](#_Toc134637048)

[4.2. Предполагаемая потребность 26](#_Toc134637049)

[4.3. Экономические преимущества разработки по сравнению с аналогами 26](#_Toc134637050)

[ПРИЛОЖЕНИЕ 1 26](#_Toc134637051)

[5. CПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 26](#_Toc134637052)

[ПРИЛОЖЕНИЕ 2 27](#_Toc134637053)

[ПРИЛОЖЕНИЕ 3 29](#_Toc134637054)

[ПРИЛОЖЕНИЕ 4 30](#_Toc134637055)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 36](#_Toc134637056)

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы: «Программа-ассистент для гостиничного бизнеса»

(«Assistance-program For Hotel Business»).

Краткое наименование программы: «HotelAssist»

## Краткая характеристика области применения программы

Программа рассчитана на использование персоналом гостиниц и отелей в служебных целях.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

## Функциональное назначение

Данное приложение предназначено для обозначения проделанной работы сотрудниками бизнес-индустрии, в которой приложение будет использовано.

## Эксплуатационное назначение

Программа будет полезна для оптимизации работы сотрудников и отслеживания проделанной работы, оно предоставляет свой функционал следующим рабочим единицам гостиницы:

1. Горничные
2. Бармены/официанты
3. Администраторы

Приложение предназначено для пользователей, работающих в гостиничной сфере бизнеса, на ПК.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Целью разработки данной программы является создание десктоп приложения для платформы Windows, которое будет предоставлять сотрудникам гостиниц инструменты для создания отчета о проделанной работе.

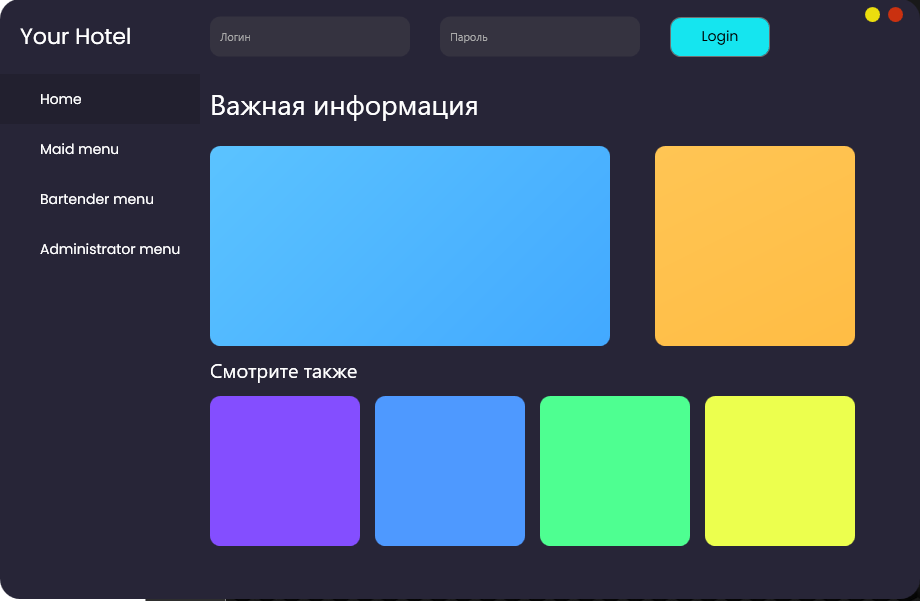
* + 1. Задачи работы

1. Продумывание архитектуры приложения, основных компонентов, а также их связей и взаимодействий.
2. Создание основного окна с возможностью переключения между страницами.
3. Создание полей для авторизации, использующих персональные логины и пароли рабочих.
4. Реализация отдельной страницы, для каждой единицы персонала, имеющей свой уникальный функционал, необходимый соответствующему работнику.
5. Проработка всех ошибочных сценариев, при которых программа, получившая неверные/ неполные входные данные, не будет аварийно закрываться.
6. Тестирование и отладка приложения, обеспечивающая стабильную работу программы.

## Описание алгоритма и функционирования программы

* + 1. **Структура приложения**

1. **Главный экран** – отображает поля для логина и пароля, сразу имеет открытое окно «Home», а также кнопки перехода по окнам для персонала.



Главный экран содержит StackPanel, в котором находятся 4 RadioButton для переключения между окнами:

<StackPanel Grid.Row="1">

<RadioButton Content="Home"

Height="50"

Foreground="White"

FontSize="14"

Style="{StaticResource MenuButtonTheme}"

IsChecked="True"

Command="{Binding HomeViewCommand}">

</RadioButton>

<RadioButton Content="Maid menu"

Height="50"

Foreground="White"

FontSize="14"

Style="{StaticResource MenuButtonTheme}"

Command="{Binding MaidViewCommand}"/>

<RadioButton Content="Bartender menu"

Height="50"

Foreground="White"

FontSize="14"

Style="{StaticResource MenuButtonTheme}"

Command="{Binding BarViewCommand}"/>

<RadioButton Content="Administrator menu"

Height="50"

Foreground="White"

FontSize="14"

Style="{StaticResource MenuButtonTheme}"

Command="{Binding AdminViewCommand}"/>

</StackPanel>

Каждая RadioButton имеет binding со своей кнопкой, которая переключает связанные View и ViewModel:

\*На примере HomeView\*

public RelayCommand HomeViewCommand { get; set; }

public HomeViewModel HomeVM { get; set; }

//Конструктор

HomeVM = new HomeViewModel();

CurrentView = HomeVM;

HomeViewCommand = new RelayCommand(o =>

{

CurrentView = HomeVM;

});

Также имеется TextBlock, содержащий название гостиницы, в нашем случае «Your Hotel»:

<TextBlock Text="Your Hotel"

VerticalAlignment="Center"

HorizontalAlignment="Left"

Foreground="White"

FontSize="22"

Margin="20,0,0,0"

FontFamily="/Fonts/#Poppins"/>

Имеются две Button:

<Button Style="{DynamicResource RoundExit}" Grid.Column="1"

Background = "#CA3110"

BorderBrush="#AA2A0E"

Height="15"

Width="15"

Margin="670,0,0,45"

Command="{Binding CloseApp}">

</Button>

<Button Style="{DynamicResource RoundMinimize}" Grid.Column="1"

Background="#EBDE0E"

Height="15"

Width="15"

Margin="625,0,0,45"

Command="{Binding MinimizeApp}">

</Button>

Каждая из которых имеет Binding с соответствующей логикой выполнения, в MainViewModel:

public ICommand CloseApp { get; set; }

public ICommand MinimizeApp { get; set; }

private void CloseAppClick()

{

App.Current.MainWindow.Close();

}

private void MinimizeAppClick()

{

App.Current.MainWindow.WindowState = WindowState.Minimized;

}

//Конструктор

CloseApp = new RelayCommand(o => CloseAppClick());

MinimizeApp = new RelayCommand(o => MinimizeAppClick());

Также содержатся два TextBox:

<TextBox Grid.Row="0"

Grid.Column="1"

HorizontalAlignment="Left"

VerticalAlignment="Center"

Margin="10,0,0,0"

Style="{StaticResource LoginTextBox}">

</TextBox>

<TextBox Grid.Row="0"

Grid.Column="1"

HorizontalAlignment="Left"

VerticalAlignment="Center"

Margin="240,0,0,0"

Style="{StaticResource PasswordTextBox}" TextChanged="TextBox\_TextChanged">

</TextBox>

Отвечающие за введенные логин и пароль.

И наконец Button:

<Button Style="{DynamicResource RoundButton}" Content="Login"

FontFamily="/Fonts/#Poppins"

FontSize="14"

Grid.Column="1"

Width="100"

Height="40"

HorizontalAlignment="Left"

VerticalAlignment="Center"

Margin="470,0,0,0"

Background="#15E5EF"

Foreground="Black"

x:Name="Login"

Command="{Binding ClickLogin}"

/>

Которая имеет логику авторизации:

private void MainButtonClick()

{

user = Logger.Login(UserName, Password);

Password = string.Empty;

UserName = string.Empty;

switch ((int)user.getPermission())

{

case 0:

break;

case 1:

ImageVisibility = Visibility.Visible;

Source = "/Images/maid.png";

IDText = "ID: " + user.getId();

break;

case 2:

ImageVisibility = Visibility.Visible;

Source = "/Images/bar.png";

IDText = "ID: " + user.getId();

break;

case 3:

ImageVisibility = Visibility.Visible;

Source = "/Images/book.png";

IDText = "ID: " + user.getId();

break;

case 4:

ImageVisibility = Visibility.Visible;

Source = "/Images/admin.png";

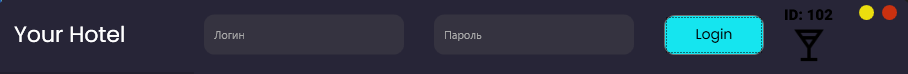
IDText = "ID: " + user.getId();

break;

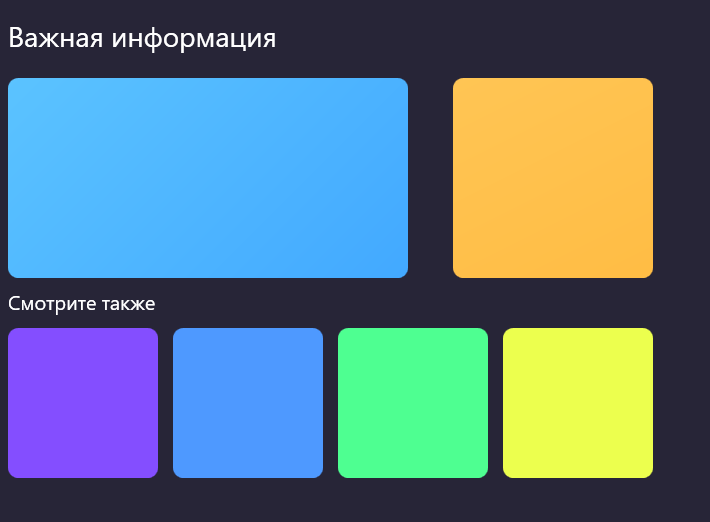
}

}

При успешном входе в аккаунт, у пользователя сверху появляется его ID и символ его работы. \*На примере Bartender



1. **Окно «Home»**



Данное окно содержит два TextBlock:

<TextBlock Text="Важная информация"

Foreground="White"

FontSize="28"

FontFamily="/Fonts/#Poppins"

HorizontalAlignment="Left"

Margin="0,0,0,20"/>

<TextBlock Text="Смотрите также"

Foreground="White"

FontFamily="/Fonts/#Poppins"

FontSize="20"

HorizontalAlignment="Left"

Margin="0,0,0,10"/>

А также Border, на месте которых в будущем, может появится любая необходимая менеджеру информация

\*Пример Border

<Border Width="400"

Height="200">

<Border.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,2">

<GradientStop Color="#5bc3ff" Offset="0.0"/>

<GradientStop Color="#3aa0ff" Offset="1"/>

</LinearGradientBrush>

</Border.Background>

<Border.Clip>

<RectangleGeometry RadiusX="10"

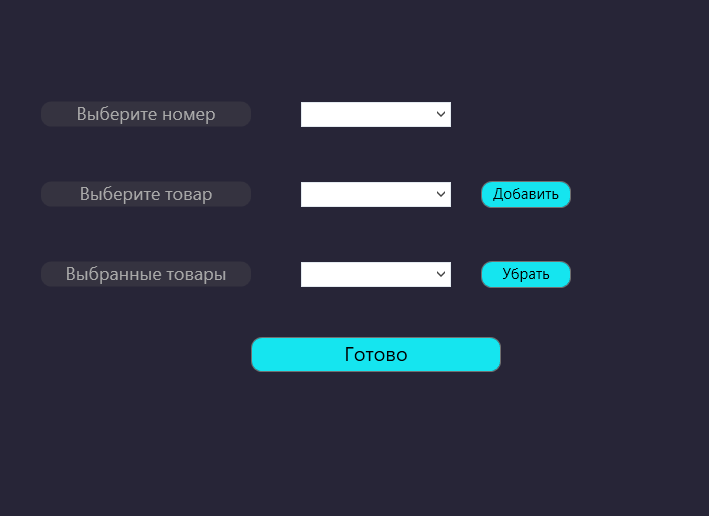
RadiusY="10"

Rect="0,0,400,200"/>

</Border.Clip>

</Border>

1. **Окно «Maid menu»**



Данное окно содержит три Border с TextBlock \*Пример:

<Border Background="#353340" Width="210" Height="25"

CornerRadius="10" Margin="20,0,0,0">

<TextBlock Text="Выберите номер" HorizontalAlignment="Center"

Foreground="DarkGray"

VerticalAlignment="Center" FontSize="18" FontFamily="/Fonts/#Poppins"/>

</Border>

Три ComboBox, каждый из которых содержит либо занятые номера в отеле, доступные и выбранные товары из мини-бара

\*На примере первого:

<ComboBox Style="{DynamicResource ComboBoxFlatStyle}" Width="150"

Height="25"

Margin="50,0,0,0"

ItemsSource="{Binding Rooms, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"

SelectedItem="{Binding SelectedRoom}"/>

Источник информации для ComboBox:

private ObservableCollection<string> rooms = new ObservableCollection<string>();

public ObservableCollection<string> Rooms

{

get { return rooms; }

set { rooms = value; }

}

Три Button, имеющие логики добавления товара в список выбранных, удаления, и кнопка завершения работы над текущим номером, после нажатия на которую, выбранный номер будет удален из списка на уборку

\*Пример Button

<Button Width="90" Height="27" Margin="30,0,0,0"

Background="#15E5EF"

Foreground="Black"

Style="{DynamicResource RoundButton}"

Content="Добавить"

FontSize="15"

FontFamily="/Fonts/#Poppins"

Command="{Binding AddProduct}"/>

AddProduct:

public ICommand AddProduct { get; set; }

private void AddProductClick()

{

if (selectedProduct != String.Empty)

{

chosenProducts.Add(selectedProduct);

SelectedProduct = null;

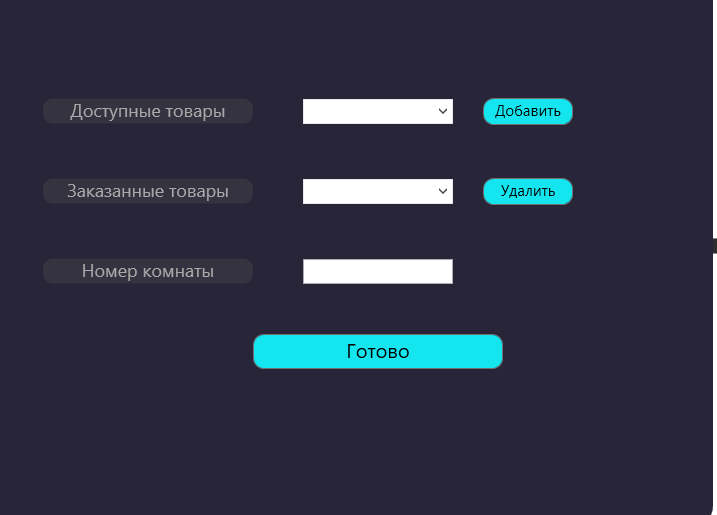
OnPropertyChanged(nameof(SelectedProduct));

}

}

AddProduct = new RelayCommand(o => AddProductClick());

1. **Окно «Bartender menu»**



Содержит похожие Border с TextBlock, а также Button и Combobox, что и в Maid Menu, однако также содержит TextBox, в который необходимо вводить номер комнаты:

<TextBox Width="150" Height="25" Margin="50,0,0,0"

Text ="{Binding Room}"

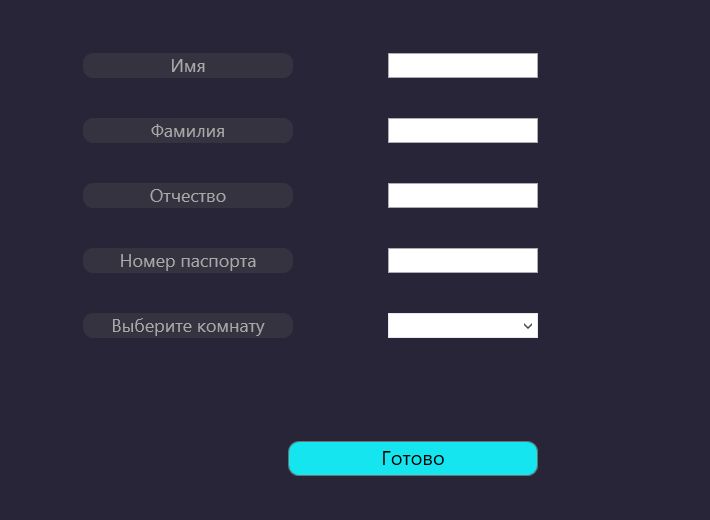
FontSize="17"

VerticalAlignment="Center"

TextAlignment="Center"

HorizontalAlignment="Center"/>

1. **Окно «Administrator Menu»**



Содержит четыре TextBox, пять Border с TextBlock, а также Button. Все поля обязательны к заполнению, за исключением «Отчество»

**3.2.2 Описание входных данных**

Всего есть 4 вида входных данных:

1. Данные о пользователях
2. Данные о баре
3. Данные о мини-баре
4. Данные о комнатах

**3.2.2.1**

Данные о пользователях находятся в текстовом документе staff.txt

Они должны быть представлены в следующем виде:

Login;password;id;permission

Пример входных данных:

maid;qwerty;101;1

bar;abc123;102;2

book;password;103;3

boss;iamtheboss;104;4

В случае, если с чтением файла произошла какая-либо ошибка, то программа выведет соответствующий MessageBox с описанием ошибки:

“Error loading rooms”

**3.2.2.2**

Данные о баре содержатся в файле bar.txt

Формат данных является следующим:

Name\_of\_product;price

Например:

Coca-Cola;150

J7 apple;120

J7 orange;120

Sprite;150

Fanta;150

Jack Daniels;400

Bacardi;350

Macallan 12;600

Macallan 18;800

Margarita;450

Long Island;650

Daiquri;500

Monaco;500

Beer;300

Chicker sandwich;400

Beef sandwich;450

Club sandwich;450

Nuts;150

В случае возникновения каких-либо проблем, система покажет MessageBox со следующим текстом:

“Error loading bar”

**3.2.2.3**

Информация о мини-баре располагается в mini-bar.txt

Формат входных данных следюущий:

Name\_of\_product

Example:

Coca-Cola 0.33

Jack Daniels 0.2

J7 apple 0.33

J7 orange 0.33

Bacardi 0.25

Sprite 0.33

Fanta 0.33

Nuts

Snickers

Mars

В тех ситуациях, когда происходит какой-либо сбой, вылезает следующее сообщение:

“Error loading mini-bar”

**3.2.2.4**

Данные о комнатах находятся в текстовом файле rooms.txt

Формат:

Number\_of\_room;IsTaken

Пример:

101;0

102;0

103;1

104;1

105;1

201;1

202;0

203;1

204;1

205;0

**3.2.3 Описание выходных данных**

Выходные данные каждого работника выгружаются в отдельный файл.

Администратор выгружает свои данные в текстовом документе с названием date\_room#number.txt

Они имеют следующий вид:

Full date

Room #..

Administator id: ..

Client information:

Name = ..

Surname = ..

Thirdname = ..

Passport number = ..

Горничная выгружает свои данные в файлы с названием date\_room\_number.txt

Формат:

date

Room #...

Maid id: …

Used mini-bar:

…

Бармен:

04.05.2023 0:54:55

Room #..

Bartender id: ..

Ordered products:

Product\_name : price

Total price: total\_price

**3.2.4 Описание алгоритма работы программы**

Первоначально, пользователь, используя свой личный логин и пароль должен войти в систему под ними, после чего он может перейти в свое окно для начала работы.

Горничная:

Для горничной существует три окна:

1. Выберите номер
2. Выберите товар
3. Выбранные товары

После загрузки входных данных, у горничной появляется список занятых номеров, которые нужно убрать.

Без выбранного номера, пользователю не получится отправить отчет об уборке номера.

После того, как был выбран номер, необходимо выбрать товары, употребленные в мини-баре. Они будут добавлены в список выбранных товаров.

public ICommand AddProduct { get; set; }

private void AddProductClick()

{

if (selectedProduct != String.Empty)

{

chosenProducts.Add(selectedProduct);

SelectedProduct = null;

OnPropertyChanged(nameof(SelectedProduct));

}

}

Если какой-то товар был добавлен по ошибке, то у пользователя есть возможность удалить лишний товар.

public ICommand DeleteProduct { get; set; }

private void DeleteProductClick()

{

if (selectedChosenProduct != String.Empty)

{

chosenProducts.Remove(selectedChosenProduct);

SelectedChosenProduct = null;

OnPropertyChanged(nameof(SelectedChosenProduct));

}

}

После завершения работы над номером, происходит нажатие кнопки готово, и данные о номере выгружаются в папку Output:

private void DoneClick()

{

if (MainVM.user.getPermission() == Permission.HouseMaid || MainVM.user.getPermission() == Permission.Manager)

{

if (selectedRoom == null)

{

MessageBox.Show("Select room");

return;

}

string context = CollectData();

string date = DateTime.Today.ToString()[..^8];

string fileName = Directory.GetCurrentDirectory() + "/OutputData/Maid/" + date + "\_room#" + selectedRoom + ".txt";

using (FileStream fs = File.Create(fileName))

{

Byte[] text = new UTF8Encoding(true).GetBytes(context);

fs.Write(text, 0, text.Length);

}

rooms.Remove(selectedRoom);

SelectedRoom = null;

OnPropertyChanged(nameof(SelectedRoom));

}

else

{

MessageBox.Show("Not enough permissions to do it");

}

}

private string CollectData()

{

string res = "";

res += DateTime.Now.ToString() + "\n";

res += "Room #" + selectedRoom + "\n";

res += "Maid id: " + MainVM.user.getId().ToString() + "\n";

res += "Used mini-bar:\n";

if (chosenProducts.Count == 0)

{

res += "none";

} else {

foreach (string a in chosenProducts)

{

res += a + "\n";

}

}

return res;

}

Бармен:

У Бармена присутствует также три окна:

1. Введите номер
2. Выберите товар
3. Выбранные товары

Изначально бармен должен вручную ввести номер посетителя, после чего добавить заказанные товары, и в случае, если он ошибся, он может их удалить из выбранных товаров.

private void AddProductClick()

{

if (selectedBar.Name != String.Empty)

{

chosenProducts.Add(selectedBar);

SelectedBar = null;

OnPropertyChanged(nameof(SelectedBar));

}

}

public ICommand DeleteProduct { get; set; }

private void DeleteProductClick()

{

if (selectedProduct.Name != String.Empty)

{

chosenProducts.Remove(selectedProduct);

SelectedProduct = null;

OnPropertyChanged(nameof(SelectedProduct));

}

}

После окончания обработки клиента, бармен должен нажать кнопку готово и отчет с ценой каждого товара и общей стоимостью заказа будет выгружен.

private void DoneClick()

{

if (MainVM.user.getPermission() == Permission.Bartender || MainVM.user.getPermission() == Permission.Manager)

{

if (Room == null)

{

MessageBox.Show("Enter room number");

return;

}

int number;

if (!int.TryParse(Room, out number))

{

MessageBox.Show("Mistake in room number");

return;

}

if (chosenProducts.Count == 0)

{

MessageBox.Show("You haven't chosen any product");

return;

}

string context = CollectData();

string date = DateTime.Today.ToString()[..^8];

string fileName = Directory.GetCurrentDirectory() + "/OutputData/Bartender/" + date + "\_room#" + Room + ".txt";

using (FileStream fs = File.Create(fileName))

{

Byte[] text = new UTF8Encoding(true).GetBytes(context);

fs.Write(text, 0, text.Length);

}

Room = null;

OnPropertyChanged(nameof(Room));

}

else

{

MessageBox.Show("Not enough permissions to do it");

}

}

private string CollectData()

{

string res = "";

res += DateTime.Now.ToString() + "\n";

res += "Room #" + Room + "\n";

res += "Bartender id: " + MainVM.user.getId().ToString() + "\n";

res += "Ordered products:\n\n";

int price = 0;

foreach (BarProduct prod in chosenProducts)

{

res += prod.Name + " : " + prod.Price + "\n";

price += prod.Price;

}

res += "\nTotal price: " + price.ToString();

return res;

}

Администратор:

У администратора присутствует 5 окон:

1. Имя
2. Фамилия
3. Отчество
4. Номер паспорта
5. Номер комнаты

В список номеров комнат загружаются все свободные, в которые может заселиться клиент.

Поля «Имя», «Фамилия» и «Номер паспорта» являются обязательными для заполнения. Поле «Отчество» не является таковым.

После ввода всей необходимой информации и нажатия кнопки готово, все данные выгружаются в соответствующий выходной файл.

private void DoneClick()

{

if (MainVM.user.getPermission() == Permission.Administrator || MainVM.user.getPermission() == Permission.Manager)

{

if (selectedRoom == null || SurName == null || Name == null || Passport == null)

{

MessageBox.Show("Enter full information");

return;

}

string context = CollectData();

string date = DateTime.Today.ToString()[..^8];

string fileName = Directory.GetCurrentDirectory() + "/OutputData/Administrator/" + date + "\_room#" + selectedRoom + ".txt";

using (FileStream fs = File.Create(fileName))

{

Byte[] text = new UTF8Encoding(true).GetBytes(context);

fs.Write(text, 0, text.Length);

}

rooms.Remove(selectedRoom);

SelectedRoom = null;

OnPropertyChanged(nameof(SelectedRoom));

Name = null;

OnPropertyChanged(nameof(Name));

SurName = null;

OnPropertyChanged(nameof(SurName));

ThirdName = null;

OnPropertyChanged(nameof(ThirdName));

Passport = null;

OnPropertyChanged(nameof(Passport));

}

else

{

MessageBox.Show("Not enough permissions to do it");

}

}

private string CollectData()

{

string res = "";

res += DateTime.Now.ToString() + "\n";

res += "Room #" + selectedRoom + "\n";

res += "Administator id: " + MainVM.user.getId().ToString() + "\n";

res += "Client information:\n";

res += "Name = " + Name + "\n";

res += "Surname = " + SurName + "\n";

res += "Thirdname = " + ThirdName + "\n";

res += "Passport number = " + Passport + "\n";

return res;

}

* 1. **Описание и обоснование выбора состава технических и программных средств**
     1. **Состав технических и программных средств**

Десктоп приложение с ОС Windows 10 и выше.

* + 1. **Обоснование выбора технических и программных средств**

Приложение разрабатывалось под оптимальную версию Windows 10, поскольку данная версия является самой распространённой для десктоп приложений

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Ориентировочная экономическая эффективность

В рамках данной работы расчет экономической эффективности не предусмотрен.

## Предполагаемая потребность

Данное приложение может быть потребно гостиницам, развитие которых находится на начальном этапе, поскольку оно способно заметно облегчить работу менеджерам, за счет того, что все отчеты о проделанной работе персонала будут находится в одном месте и иметь собранный и единый вид.

## Экономические преимущества разработки по сравнению с аналогами

Сравнение программного продукта “Мобильный сервис ресторанного меню на основе шаблонов с аналогами представлено в Таблице 1.

Таблица 1. Сравнение с аналогами.

|  |  |  |
| --- | --- | --- |
| **Название** | **Ссылка** | **Сравнение** |
| Maxi Booking | https://maxi-booking.ru/programma-dlya-otelej-i-gostinits/ | Недостатки: цена, отсутствие возможности отслеживания работы некоторого персонала  Преимущества: отслеживание загрузки отеля, инструменты для аналитики |
| PMS Shelter | https://shelter.ru/cloud/ | Недостатки: функционал ограничивается заселением клиентов, цена  Преимущества: Интеграции с другими приложениями |

# ПРИЛОЖЕНИЕ 1

# CПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.301-79 Программа и методика испытаний. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.401-78 Текст программы. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.505-79 Руководство оператора. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.404-79 Пояснительная записка. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. Википедия: свободная энциклопедия [Электронный ресурс]. // URL: <https://ru.wikipedia.org> (Дата обращения: 06.05.2023, режим доступа: свободный)

# ПРИЛОЖЕНИЕ 2

**ТЕРМИНОЛОГИЯ**

Таблица 2

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| SSD/HDD | Тип диска для хранения памяти на устройстве. |
| Шаблон | Форма документа с уже заполненными полями, которые можно использовать как пример. |
| Программа-ассистент | Вспомогательная программа, упрощающая часть задач |
| Окно входа | Окно, в котором пользователю предоставляется ввести свой логин и пароль. |

# ПРИЛОЖЕНИЕ 3

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

|  |  |
| --- | --- |
| Класс | Назначение |
| ObservableObject | Класс, автоматически инициирующий PropertyChanged |
| RelayCommand | Класс, активирующий определенную функцию |
| BarProduct | Товар бара |
| Logger | Класс, реализующий работу с пользователями |
| User | Пользователь |
| AdminViewModel | ViewModel окна администратора |
| BarViewModel | ViewModel окна бармена |
| MaidViewModel | ViewModel окна горничной |
| HomeViewModel | ViewModel окна «Home» |
| MainViewModel | ViewModel основного окна |

# ПРИЛОЖЕНИЕ 4

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ И ФУНКЦИЙ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** |  |  |  |  |
| **Класс** | **ObservableObject** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **OnPropertyChanged** | **protected** | **void** | **String name** | **Оповещение об изменении состояния объекта** |
| **Класс** | **Logger** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **clearData** | **public** | **void** | **-** | **Очищение данных** |
| **LoadData** | **public** | **void** | **-** | **Загрузка данных из файла** |
| **Класс** | **User** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **Log** | **public** | **void** | **Int id, Permission permission** | **Вход в аккаунт** |
| **getId** | **public** | **int** | **-** | **Return id;** |
| **getPermission** | **public** | **Permission** | **-** | **Return Permission;** |
| **Класс** | **AdminViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **loadRooms** | **private** | **void** | **-** | **Загрузка списка комнат** |
| **DoneClick** | **private** | **void** | **-** | **Выгрузка информации в .txt** |
| **CollectData** | **Private** | **string** | **-** | **Сбор и организация данных для вывода** |
| **Класс** | **BarViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **loadBar** | **private** | **void** | **-** | **Загрузка бара** |
| **AddProductClick** | **private** | **void** | **-** | **Добавление продукта** |
| **DeleteProductClick** | **private** | **void** | **-** | **Удаление продукта из списка выбранных** |
| **DoneClick** | **private** | **void** | **-** | **Выгрузка информации в .txt** |
| **CollectData** | **private** | **string** | **-** | **Сбор и организация информации** |
| **Класс** | **MaidViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **loadRooms** | **private** | **void** | **-** | **Загрузка комнат** |
| **loadProducts** | **private** | **Void** | **-** | **Загрузка продуктов** |
| **AddProductClick** | **private** | **void** | **-** | **Добавление продукта** |
| **DeleteProductClick** | **private** | **void** | **-** | **Удаление продукта из списка выбранных** |
| **DoneClick** | **private** | **void** | **-** | **Выгрузка информации в .txt** |
| **CollectData** | **private** | **string** | **-** | **Сбор и организация информации** |
| **Класс** | **MainViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **MainButtonClick** | **private** | **Void** | **-** | **Вход в аккаунт** |
| **CloseAppClick** | **private** | **void** | **-** | **Закрытие приложения** |
| **MinimizeAppClick** | **private** | **void** | **-** | **Скрытие приложения** |
| **Поля и свойства** |  |  |  |  |
| **Класс** | **BarProduct** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **name** | **private** | **string** | **Get; set;** | **Название продукта** |
| **price** | **private** | **int** | **Get; set;** | **Цена продукта** |
| **Name** | **public** | **string** | **Get;** | **Возвращает название** |
| **Price** | **public** | **int** | **Get;** | **Возвращает цену** |
| **Класс** | **Logger** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **rootFolder** | **Private readonly** | **string** | **-** | **Источник данных** |
| **people** | **private** | **List<Person>** | **-** | **Список возможных пользователей** |
| **Класс** | **User** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **isLogged** | **public** | **bool** | **Get; set;** | **Вошел ли пользователь** |
| **id** | **private** | **int** | **Get; set;** | **id** |
| **permission** | **private** | **Permission** | **Get; set;** | **Должность** |
| **Класс** | **AdminViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **MainVM** | **private** | **MainViewModel** | **-** | **Основной ViewModel** |
| **rooms** | **private** | **ObservableCollection <string>** | **-** | **Список доступных комнат** |
| **roomsFolder** | **Private readonly** | **string** |  | **Источник комнат** |
| **Name** | **public** | **string** | **Get; set;** | **Имя** |
| **SurName** | **public** | **string** | **Get; set;** | **Фамилия** |
| **ThirdName** | **public** | **string** | **Get; set;** | **Отчество** |
| **Passport** | **public** | **string** | **Get; set;** | **Номер паспорта** |
| **SelectedRoom** | **public** | **string** | **Get; set;** | **Выбранная комната** |
| **Done** | **public** | **ICommand** | **Get; set;** | **Нажатие кнопки** |
| **Класс** | **BarViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **MainVM** | **private** | **MainViewModel** | **-** | **Основной ViewModel** |
| **bar** | **private** | **IList**  **<BarProduct>** |  | **Список доступных товаров** |
| **chosenProducts** | **private** | **ObservableCollection**  **<BarProduct>** |  | **Список выбранных продуктов** |
| **SelectedBar** | **public** | **BarProduct** | **Get; set;** | **Выбранный товар** |
| **SelectedProduct** | **Public** | **BarProduct** | **Get; set;** | **Выбранный товар для удаления** |
| **AddProduct** | **public** | **ICommand** | **Get; set;** | **Кнопка добавления** |
| **DeleteProduct** | **public** | **ICommand** | **Get; set;** | **Кнопка удаления** |
| **Room** | **public** | **string** | **Get; set;** | **Комната** |
| **Done** | **public** | **ICommand** | **Get; set;** | **Кнопка готово** |
| **Класс** | **MaidViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **MainVM** | **private** | **MainViewModel** | **-** | **Основной ViewModel** |
| **chosenProducts** | **private** | **ObservableCollection**  **<BarProduct>** |  | **Список выбранных продуктов** |
| **rooms** | **private** | **ObservableCollection <string>** | **-** | **Список доступных комнат** |
| **roomsFolder** | **Private readonly** | **string** |  | **Источник комнат** |
| **chosenProducts** | **private** | **ObservableCollection**  **<BarProduct>** |  | **Список выбранных продуктов** |
| **SelectedProduct** | **Public** | **BarProduct** | **Get; set;** | **Выбранный товар для удаления** |
| **SelectedRoom** | **public** | **string** | **Get; set;** | **Выбранная комната** |
| **AddProduct** | **public** | **ICommand** | **Get; set;** | **Кнопка добавления** |
| **DeleteProduct** | **public** | **ICommand** | **Get; set;** | **Кнопка удаления** |
| **Done** | **public** | **ICommand** | **Get; set;** | **Кнопка готово** |
| **Класс** | **MainViewModel** |  |  |  |
| **Имя** | **Модификатор доступа** | **Тип** | **Аргументы** | **Назначение** |
| **HomeViewCommand** | **public** | **RelayCommand** | **Get; set;** | **Команда перехода на окно «Home»** |
| **MaidViewCommand** | **public** | **RelayCommand** | **Get; set;** | **Команда перехода на окно «Maid»** |
| **BarViewCommand** | **public** | **RelayCommand** | **Get; set;** | **Команда перехода на окно «Bar»** |
| **AdminViewCommand** | **public** | **RelayCommand** | **Get; set;** | **Команда перехода на окно «Admin»** |
| **user** | **public** | **User** |  | **пользователь** |
| **HomeVM** | **Public** | **HomeViewModel** | **Get; set;** | **ViewModel окна Home** |
| **MaidVM** | **Public** | **MaidViewModel** | **Get; set;** | **ViewModel окна Maid** |
| **BarVM** | **Public** | **BarViewModel** | **Get; set;** | **ViewModel окна Bar** |
| **AdminVM** | **public** | **AdminViewModel** | **Get; set** | **ViewModel окна Admin** |
| **CurrentView** | **public** | **object** | **Get; set;** | **Текущий View** |
| **UserName** | **public** | **string** | **Get; set;** | **Введённый логин** |
| **Password** | **public** | **string** | **Get; set;** | **Введенный пароль** |
| **ImageVisibility** | **public** | **Visibility** | **Get; set;** | **Видимость картинки** |
| **IDText** | **public** | **string** | **Get; set;** | **Текст id пользователя** |
| **Source** | **public** | **string** | **Get; set;** | **Источник картинки** |
| **ClickLogin** | **public** | **ICommand** | **Get; set;** | **Кнопка Login** |
| **CloseApp** | **public** | **ICommand** | **Get; set;** | **Кнопка закрытия окна** |
| **MinimizeApp** | **public** | **ICommand** | **Get; set;** | **Кнопка скрытия окна** |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |