Министерство образования и науки Российской Федерации

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСТИТЕТ

Институт информационных технологий и анализа данных

ОТЧЕТ

к лабораторной работе по дисциплине:

|  |
| --- |
| **Инструментальные средства информационных систем** |
| Построение диаграмм UML |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | ИСМб-19-1 |  |  |  | Кирилов Д.А. |
|  | шифр группы |  | подпись |  | фамилия И.О. |
| Проверил |  |  |  |  | Дородных Н.О. |
|  | должность |  | подпись |  | фамилия И.О. |

|  |  |
| --- | --- |
|  |  |

Иркутск 2021 г.

**Содержание**

[1. Постановка задачи 3](#_Toc20212065)

[2. Описание предметной области 3](#_Toc20212066)

[3. Основные модели вариантов использования 4](#_Toc20212067)

[4. Описания основных вариантов использования 4](#_Toc20212068)

[5. Системная диаграмма последовательностей действий 5](#_Toc20212076)

[6. Диаграмма деятельностей, описывающая вариант использования 6](#_Toc20212077)

[7. Диаграмма классов 7](#_Toc20212078)

[8.Фрагмент программного кода 8](#_Toc20212079)

[9.Фрагмент документации 9](#_Toc20212080)

# Постановка задачи

Для выбранного варианта задания необходимо:

1. Разработать диаграмму вариантов использования в нотации UML;
2. Выполнить описание основных вариантов использования.
3. Разработать системную диаграмму последовательностей действий;
4. Разработать диаграммы деятельностей;
5. Разработать диаграмму классов уровня проектирования в нотации UML;
6. Сгенерировать программный код на основе разработанной диаграммы классов;
7. Сгенерировать комплект документации.

# Описание предметной области

Имеются пациенты (id, ФИО, диагноз) и больницы (id, номер, адрес). Необходимо зафиксировать пациентов в больницах и дату прихода пациента.

# Основные модели вариантов использования

Основная модель вариантов использования представлена на рисунке 1.

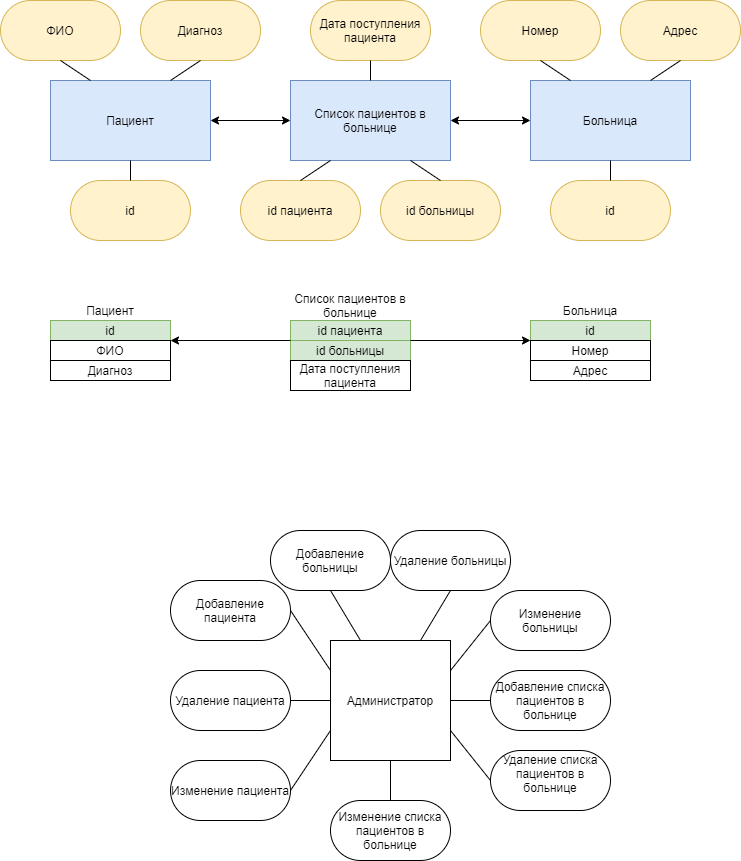


Рисунок 1 – Диаграмма вариантов использования

# Описания основных вариантов использования

Система разрабатывается для одного человека, поэтому на диаграмме всего одно действующее лицо. Все описанные функции выполняет руководитель администратор.

**Спецификация варианта использования «Добавить новую больницу в больничную сеть»**

Цель: Добавление больницы

Действующее лицо: администратор

Предусловия: Главное меню системы

**Основной сценарий**:

1. Отображение формы добавление новой больницы;
2. Администратор вводит данные об больнице: номер, адрес;
3. Проверка введённых данных;
4. Система автоматически заносит данные в хранилище;
5. Отображение новой больницы в списке больничной сети.

**Ошибка: На шаге 2**

1. Пользователь неправильно ввёл данные или не ввёл одно из полей;
2. Система возвращается на 2 шаг и выводит ошибку о некорректном вводе данных;

# 5. Системная диаграмма последовательностей действий

На рисунке 2 представлена системная диаграмма последовательностей действий для добавления новой больницы.

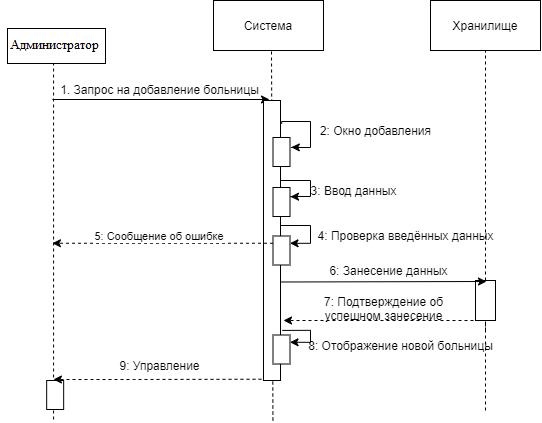


Рисунок 2 – Диаграмма последовательности добавления новой больницы

# 6. Диаграмма деятельностей, описывающая вариант использования

На рисунке 3 представлена диаграмма деятельности, описывающая вариант использования «Добавление больницы».

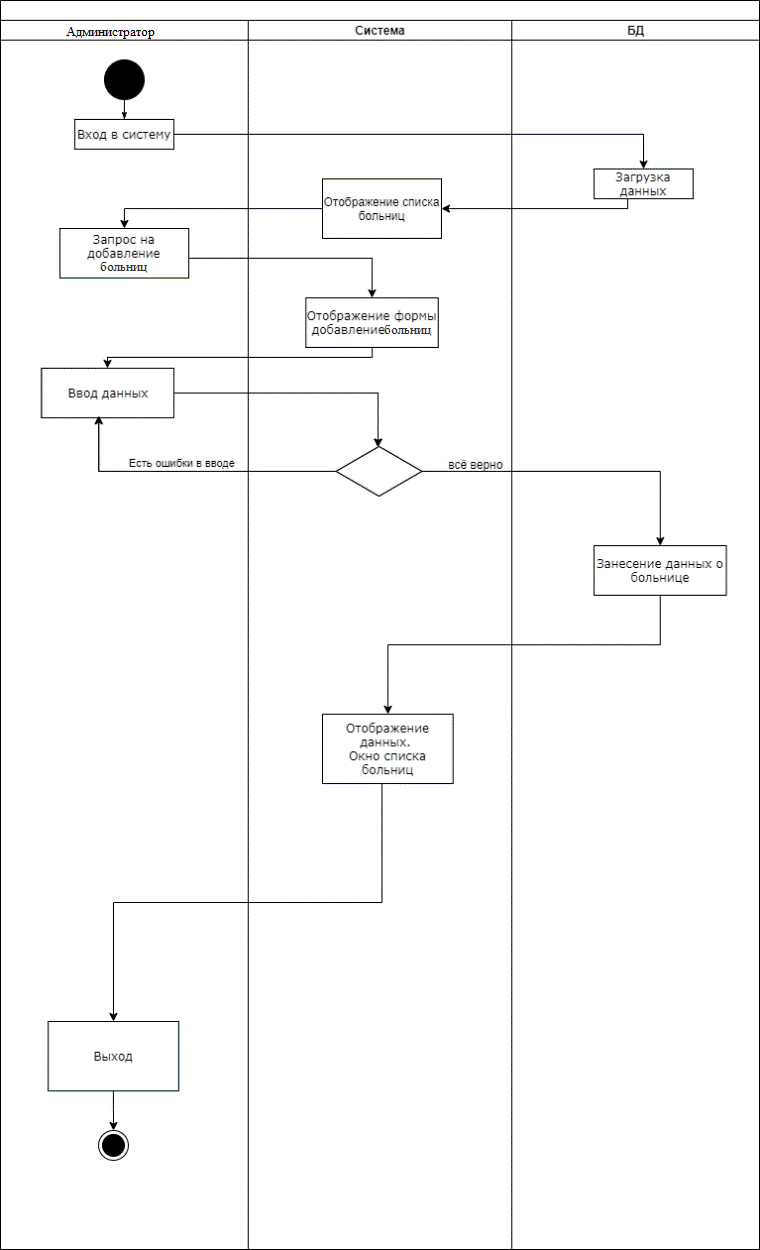


Рисунок 3 – Диаграмма активности добавление новой больницы

# 7. Диаграмма классов

На рисунке 4 представлена диаграмма классов.

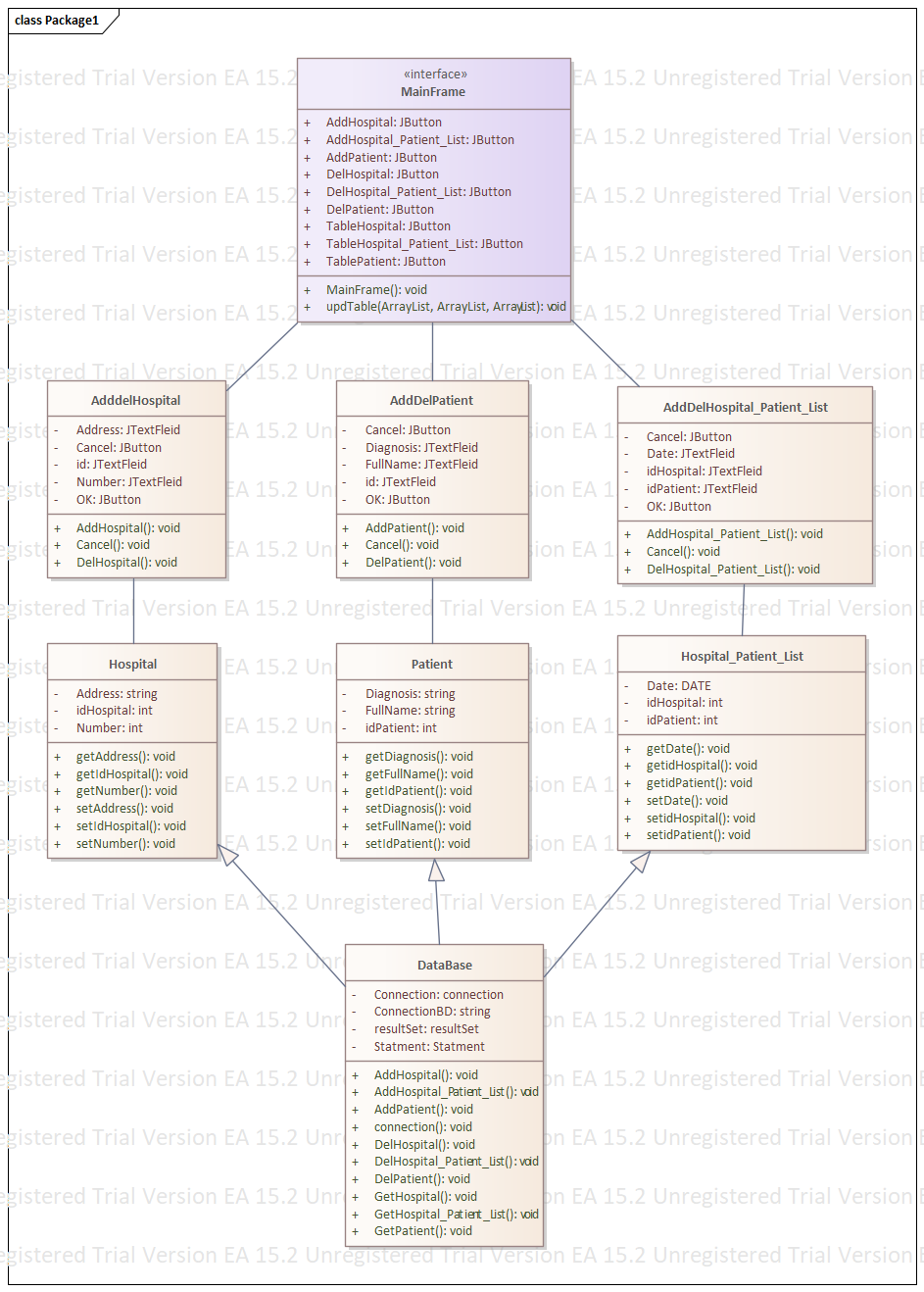


Рисунок 4 – Диаграмма классов

# 8.Фрагмент программного кода

|  |
| --- |
| package Package1;  public class AddDelPatient{  private JTextField FullName;  private JTextField Diagnosis;  private JTextField id;  private JButton OK;  private JButton Cancel;  public AddDelPatient(){  }  public void finalize() throws Throwable {  }  public void AddPatient(){  }  public void DelPatient(){  }  public void Cancel(){  }  }  public class AddDelHospital{  private JTextField Address;  private JTextField Number;  private JTextField id;  private JButton OK;  private JButton Cancel;  public AddDelHospital(){  }  public void finalize() throws Throwable {  }  public void AddHospital(){  }  public void DelHospital(){  }  public void Cancel(){  }  }  public class AddDelHospital\_Patient\_List{  private JTextField Date;  private JTextField idHospital;  private JTextField idPatient;  private JButton OK;  private JButton Cancel;  public AddDelHospital\_Patient\_List(){  }  public void finalize() throws Throwable{  }  public void AddHospital\_Patient\_List(){  }  public void DelHospital\_Patient\_List(){  }  public void Cancel(){  }  }  public class Patient{  private string FullName;  private string Diagnosis;  private int idPatient;  public Patient(){  }  public void finalize() throws Throwable {  }  public void getFullName(){  }  public void getDiagnosis(){  }  public void getIdPatient(){  }  public void setFullName(){  }  public void setDiagnosis(){  }  public void setIdPatient(){  }  }  public class Hospital{  private string Address;  private int Number;  private int idHospital;  public Hospital(){  }  public void finalize() throws Throwable {  }  public void getAddress(){  }  public void getNumber(){  }  public void getIdHospital(){  }  public void setAddress(){  }  public void setNumber(){  }  public void setIdHospital(){  }  }  public class Hospital\_Patient\_List{  private DATE Date;  private int idPatient;  private int idHospital;  public Hospital\_Patient\_List(){  }  public void finalize() throws Throwable {  }  public void getDate(){  }  public void getIdPatient(){  }  public void getIdHospital(){  }  public void setDate(){  }  public void setIdPatient(){  }  public void setIdHospital(){  }  }  public class DataBase{  private connection Connection;  private string ConnectionBD;  private resultSet resultSet;  private Statment Statment;  public DataBase(){  }  public void finalize() throws Throwable {  }  public void AddHospital(){  }  public void AddHospital\_Patient\_List(){  }  public void AddPatient(){  }  public void DelHospital(){  }  public void DelHospital\_Patient\_List(){  }  public void DelPatient(){  }  public void GetHospital(){  }  public void GetHospital\_Patient\_List(){  }  public void GetPatient(){  }  public void connection(){  }  }  public class MainFrame{  private JButton AddHospital;  private JButton AddHospital\_Patient\_List;  private JButton AddPatient;  private JButton DelHospital;  private JButton DelHospital\_Patient\_List;  private JButton DelPatient;  private JButton TableHospital;  private JButton TableHospital\_Patient\_List;  private JButton TablePatient;  public MainFrame(){  }  public void finalize() throws Throwable {  }  public void updTable(ArrayList Hospital, ArrayList Hospital\_Patient\_List, ArrayList Patient){  }  } |

# 9.Фрагмент документации

**Руководство пользователя**

**1. Введение**

1.1. Область применения

Программа «Автоматизация регистрации пациентов в больнице» далее АРПБ – комплекс, предназначенный для учета пациентов в больницах.

1.2. Краткое описание возможностей

Данный комплекс позволят упростить и автоматизировать процесс учёта пациентов в больницах.

1.3. Уровень подготовки пользователя:

Пользователь должен иметь опыт работы с ОС Windows, версия не ниже 7.

1.4. Перечень эксплуатационной документации, с которой необходимо

ознакомиться пользователю: • Руководство пользователя (user’s guide);