**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ**

**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

**Факультет компьютерных наук**

**Информационная система медицинского центра**

Яковлева Ю.А.

Воронеж 2019

**Оглавление**

[Диаграмма вариантов использования 3](#_gjdgxs)

[Диаграммы классов 6](#_30j0zll)

[Диаграмма пакетов 8](#_1fob9te)

[Диаграммы последовательности 9](#_3znysh7)

[Кооперативные диаграммы 12](#_2et92p0)

[Диаграммы состояний 14](#_tyjcwt)

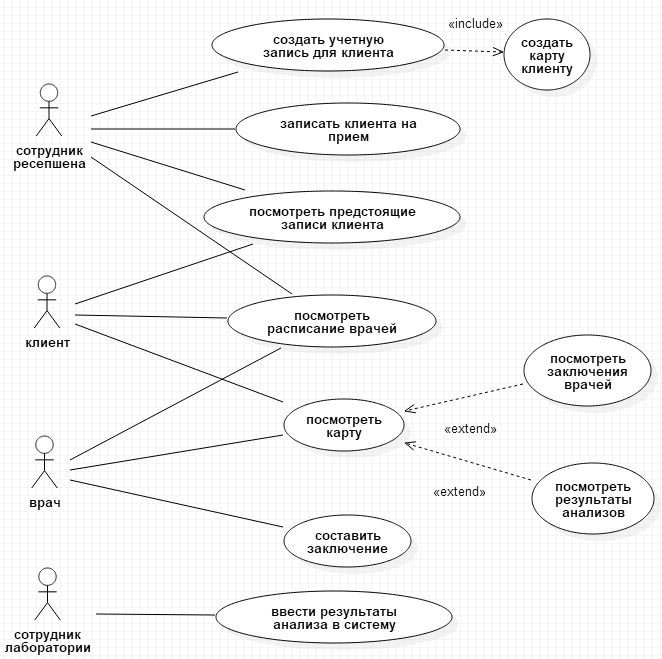
[Диаграммы деятельности 16](#_3dy6vkm)

[Диаграмма компонентов 18](#_1t3h5sf)

[Диаграмма Развертывания 19](#_4d34og8)

# Диаграмма вариантов использования

Диаграмма вариантов использования представлена на рисунке ниже:



Актёры:

* Сотрудник ресепшена - пользователь, занимающийся организацией записи клиентов. Он может создавать учётные записи для клиентов, записывать их на приём, просматривать их предстоящие записи и просматривать расписание врачей.
* Клиент - пользователь, получающий услуги в мед-центре. Он может просматривать список своих предстоящих записей, просматривать свою карту (можно посмотреть результаты анализов и заключения врачей) и смотреть расписание врачей.
* Врач - пользователь, вносящий информацию в карту клиента. Он может составить заключение по результату осмотра клиента, просмотреть всю карту клиента (в том числе результаты анализов и другие заключения врачей) и посмотреть расписание врачей.
* Сотрудник лаборатории - пользователь, вносящий информацию о проведенных анализах. Он может внести результаты анализа в систему.

Варианты использования:

* Создать учётную запись для клиента

Описание: данный вариант использования позволяет создать в системе учётную запись клиента, включая создание карты для него.

Основной поток событий:

1. Сотрудник ресепшена запрашивает данные у клиента.
2. Клиент представляет необходимые данные.
3. Сотрудник ресепшена вводит данные в форму и подтверждает операцию создания.

* Записать клиента на прием

Описание: данный вариант использования позволяет зарегистрировать в системе запись клиента на прием.

Основной поток событий:

1. Клиент сообщает сотруднику ресепшена, какого типа медицинская услуга ему необходима, и какое время записи для него удобно.
2. Если необходимый специалист в необходимое время занят, вариант использования завершается, в противном случае сотрудник ресепшена заполняет форму записи и подтверждает создание записи.

* Посмотреть предстоящие записи клиента

Описание: данный вариант использования позволяет посмотреть запланированные на будущее записи клиента.

Основной поток событий:

1. Клиент в своем личном кабинете выбирает пункт просмотра предстоящих записей.
2. Предстоящие записи отображаются.

* Посмотреть карту

Описание: данный вариант использования позволяет посмотреть карту клиента, включая результаты его анализов и заключения врачей.

Основной поток событий:

1. Клиент в своем личном кабинете выбирает пункт просмотра карты.
2. Карта отображается.

* Составить заключение

Описание: данный вариант использования позволяет врачу создать запись с заключением по результату приема.

Основной поток событий:

1. Врач в своем личном кабинете выбирает страницу с текущим приёмом.
2. Отображается форма для составления заключения
3. После осмотра пациента врач заполняет форму и подтверждает создание записи.

* Ввести результаты анализа в систему

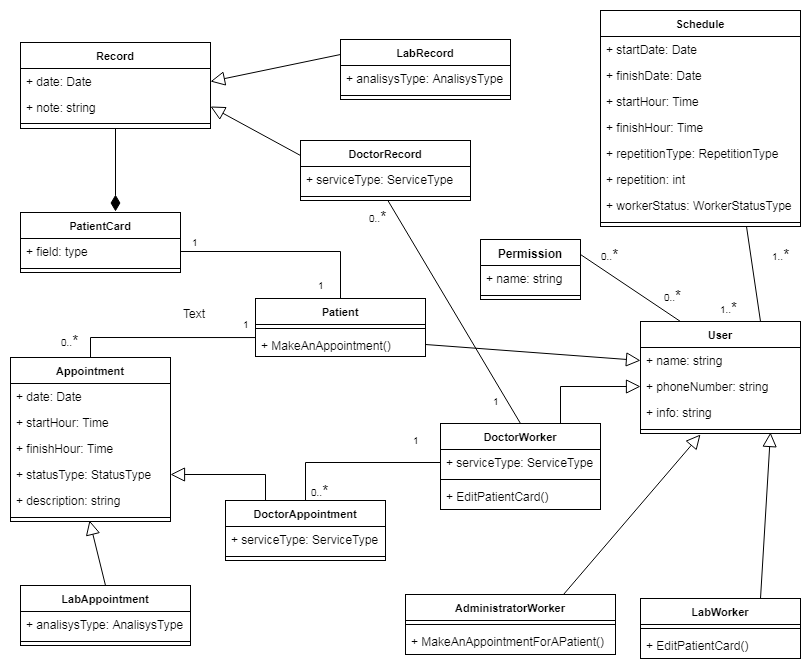
Описание: данный вариант использования позволяет работнику лаборатории создать запись о результате анализа в карте клиента.

Основной поток событий:

1. Работник лаборатории в своем личном кабинете переходит на страницу создания записи результата анализа.
2. Отображается форма создания записи о новом анализе.
3. Работник вводит данные анализа и подтверждает создание записи.

# Диаграммы классов

Диаграмма классов представлена на рисунке ниже:



Данная диаграмма отображает основные сущности и отношения между ними.

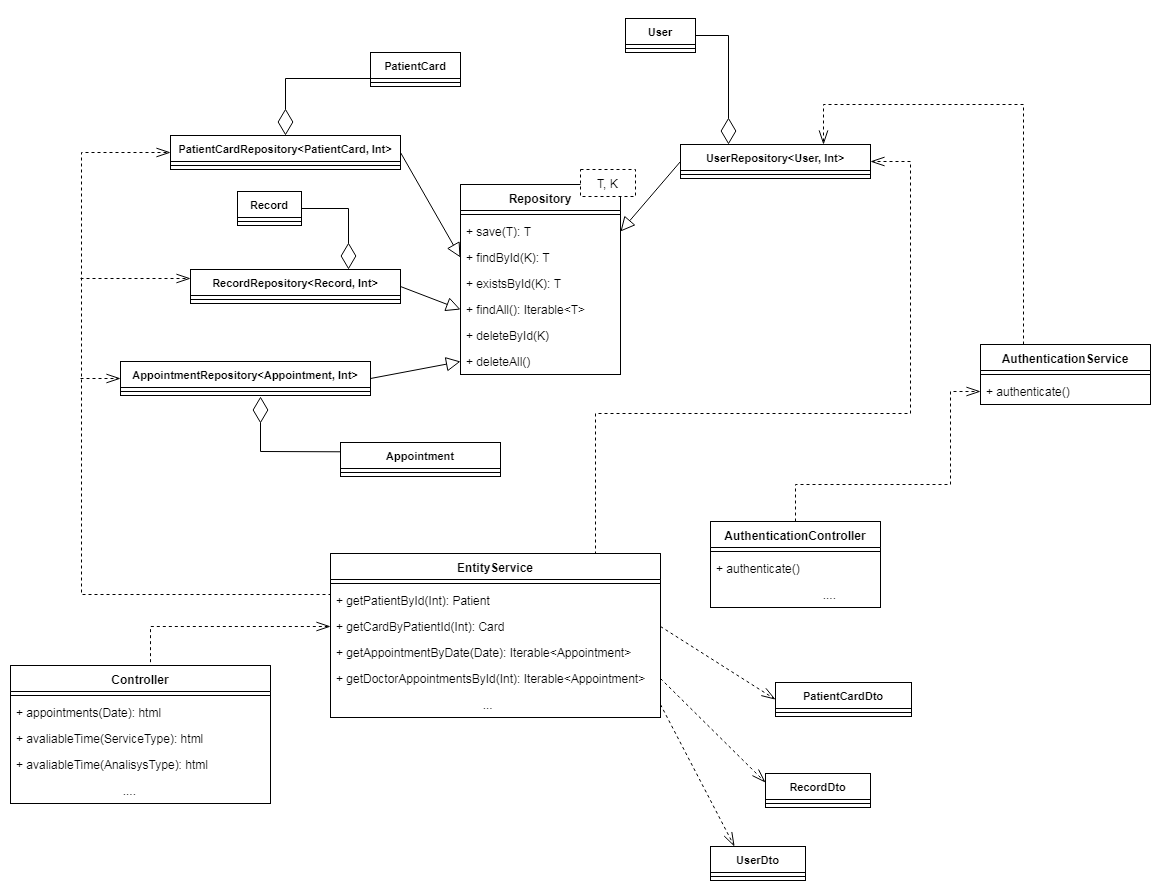
User – базовый класс для таких классов, как DoctorWorker, LabWorker, AdministratorWorker и Patient, представляющий собой сущности пользователей системы. У данных пользователей есть Permission, обозначающих их роль в системе.

Класс Schedule представляет собой расписание работников.

Класс PatientCard представляет собой карту пациента, которая включает в себя несколько экземпляров класса Record, представляющие собой записи врачей и информации об анализах в карте пациента.

Appointment является базовым классом для представления записи клиента на прием.

Диаграмма классов уровня реализации представлена на рисунке ниже:



На данной диаграмме представлена общая архитектура приложения.

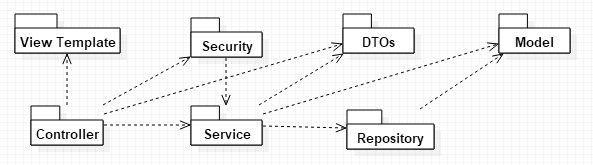
Потомки класса Repository представляют собой классы уровня доступа к базе двнных. Класс Repository является шаблоном. T – тип хранимой сущности, K – тип первичного ключа этой сущности.

EntityService и AuthenticationService обращаются к соответствующим потомкам Repository для предоставления определённого сервиса классам Controller и AuthenticationController.

Классы PatientCardDto, RecordDto и UserDto представляют собой классы, отвечающие за промежуточное представление классов PatientCard, Record и User при их отображении клиенту.

# Диаграмма пакетов

Диаграмма пакетов приложения представлена на рисунке ниже:



На диаграмме пакетов представлены следующие пакеты:

* Model - содержит все классы бизнес-логики приложения
* Repository - содержит классы-репозитории, являющиеся связывающим звеном между программой и базой данных. Классы этого пакета зависят от классов пакета Model.
* DTOs - содержит классы, являющиеся собой представлением классов бизнес-логики, содержащие только необходимые для отображения пользователю атрибуты.
* Service - содержит классы, являющиеся абстракциями над классами пакета Repository. Также пакеты этого класса отображают сущности классов пакета Model, получаемые с помощью классов пакета Repository, в сущности классов пакета DTOs.
* Security - содержит классы, необходимые для аутентификации и авторизации пользователей.
* View Template - содержит файлы шаблонизатора.
* Controller - содержит классы, определяющие набор API, с которым взаимодействует клиент приложения. Классы данного пакета логически объединяют классы пакета Service, также они взаимодействуют с классами из пакета Security и передают файлам пакета View Templates классы пакета DTOs для отображения.

# Диаграммы последовательности

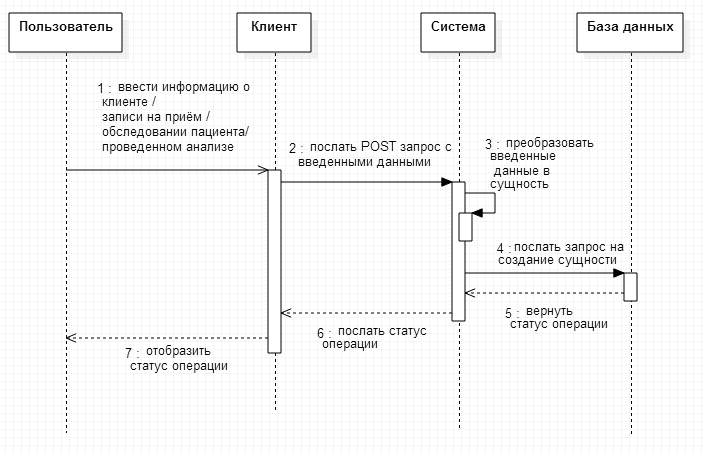
Диаграмма последовательности для таких вариантов использования, как Посмотреть предстоящие записи клиента, Посмотреть расписание врачей, Посмотреть карту пациента, Посмотреть заключения врачей, Посмотреть результаты анализов, представлена ниже: Объекты:

1. Пользователь - сотрудник ресепшена или клиент мед-центра, врач или сотрудник лаборатории.
2. Клиент - веб-браузер
3. Система
4. База данных

Сообщения:

1. Запросить информацию о предстоящих записях клиента (вариант использования Посмотреть предстоящие записи клиента)/ о расписании врачей (вариант использования Посмотреть расписание врачей)/ о карте пациента (вариант использования Посмотреть карту пациента)
2. Послать GET запрос (на определенный end point в зависимости от того, какие данные запрашиваются)
3. Послать запрос на получение сущностей (тип сущности определяется запрашиваемыми данными)
4. Вернуть соответствующие запросу сущности
5. Послать полученные сущности в удобном для пользователя представлении
6. Отобразить полученные данные

Диаграмма последовательности для таких вариантов использования, как Зарегистрировать клиента в системе, Создать учётную запись клиента, Записать клиента на приём, Составить заключение, Ввести результаты анализов в систему, представлена на рисунке ниже:



Объекты:

1. Пользователь - сотрудник ресепшена или клиент мед-центра, врач или сотрудник лаборатории.
2. Клиент - веб-браузер
3. Система
4. База данных

Сообщения:

1. Ввести информацию о клиенте (для варианта использования Зарегистрировать клиента в системе, Создать учётную запись клиента)/ о записи на приём (для варианта использования Записать клиента на прием)/ об обследовании пациента (для варианта использования Составить заключение)/ о проведенном анализе (для варианта использования Ввести результаты анализа в систему)
2. Послать POST запрос с введёнными данными
3. Преобразовать введённые данные в сущности (из сущностей для представления клиенту в сущности для обработки базой данных)
4. Послать запрос на создание сущности
5. Вернуть статус операции (добавления сущности)
6. Послать статус операции
7. Отобразить статус операции

# Кооперативные диаграммы

Диаграмма кооперации для таких вариантов использования, как Посмотреть предстоящие записи клиента, Посмотреть расписание врачей, Посмотреть карту пациента, Посмотреть заключения врачей, Посмотреть результаты анализов, представлена ниже:

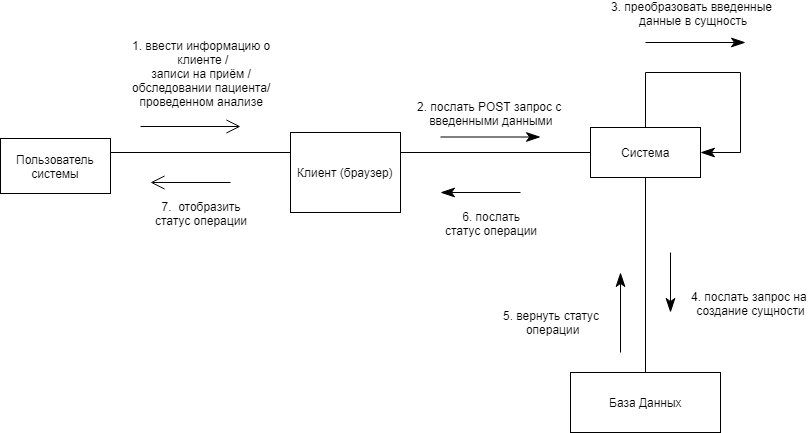
Объекты:

1. Пользователь - сотрудник ресепшена или клиент мед-центра, врач или сотрудник лаборатории.
2. Клиент - веб-браузер
3. Система
4. База данных

Сообщения:

1. Запросить информацию о предстоящих записях клиента (вариант использования Посмотреть предстоящие записи клиента)/ о расписании врачей (вариант использования Посмотреть расписание врачей)/ о карте пациента (вариант использования Посмотреть карту пациента)
2. Послать GET запрос (на определенный end point в зависимости от того, какие данные запрашиваются)
3. Послать запрос на получение сущностей (тип сущности определяется запрашиваемыми данными)
4. Вернуть соответствующие запросу сущности
5. Послать полученные сущности в удобном для пользователя представлении
6. Отобразить полученные данные

Диаграмма кооперации для таких вариантов использования, как Зарегистрировать клиента в системе, Создать учётную запись клиента, Записать клиента на приём, Составить заключение, Ввести результаты анализов в систему, представлена на рисунке ниже:



Объекты:

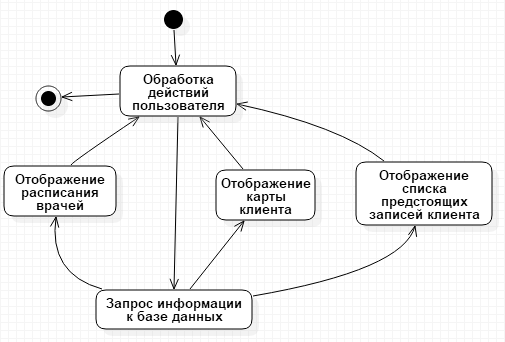
1. Пользователь - сотрудник ресепшена или клиент мед-центра, врач или сотрудник лаборатории.
2. Клиент - веб-браузер
3. Система
4. База данных

Сообщения:

1. Ввести информацию о клиенте (для варианта использования Зарегистрировать клиента в системе, Создать учётную запись клиента)/ о записи на приём (для варианта использования Записать клиента на прием)/ об обследовании пациента (для варианта использования Составить заключение)/ о проведенном анализе (для варианта использования Ввести результаты анализа в систему)
2. Послать POST запрос с введёнными данными
3. Преобразовать введённые данные в сущности (из сущностей для представления клиенту в сущности для обработки базой данных)
4. Послать запрос на создание сущности
5. Вернуть статус операции (добавления сущности)
6. Послать статус операции
7. Отобразить статус операции

# Диаграммы состояний

Диаграмма состояний системы при авторизации в ней клиента мед-центра представлена ниже.



На данной диаграмме можно наблюдать состояния системы, в которой в данный момент авторизован клиент медицинского центра. Из состояния Обработка действия пользователя система может перейти к терминальному состоянию или перейти в состояние Запроса информации (при желании клиента посмотреть свою карту, список своих запланированных записей или расписание врачей). Из состояния Запроса информации система может перейти в состояния Отображение расписания врачей, Отображение карты клиента или Отображения списка предстоящих записей в зависимости от того, запрос какого ресурса был послан. Из этих трёх состояний система переходит в состояние Обработка действия пользователя (после совершения каких-либо действий клиентом).

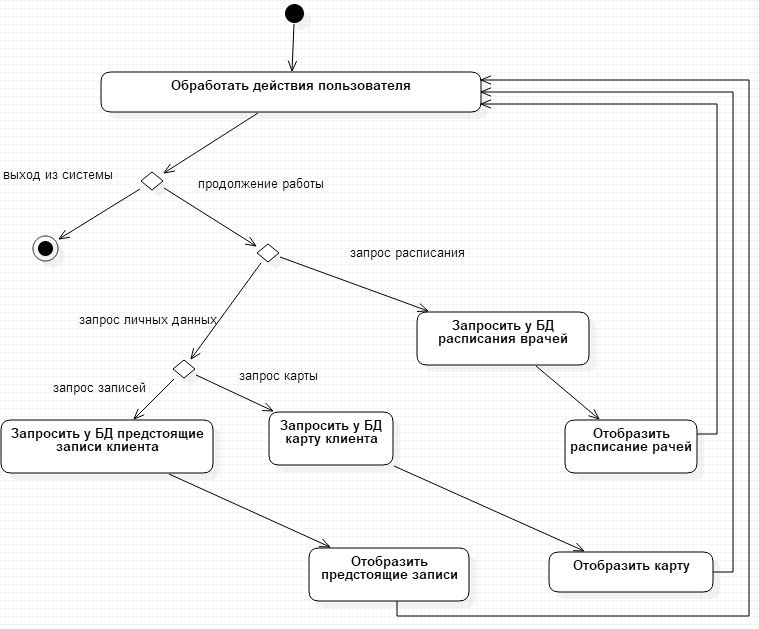
Диаграмма состояний системы при авторизации в ней сотрудника лаборатории представлена ниже:



Из состояния Обработка действия пользователя система может перейти к терминальному состоянию или перейти в состояние Отображение формы для ввода результата анализа. После ввода результатов и подтверждения система переходит в состояние Сохранение данных об анализе. После сохранения будет осуществлен переход в состояние Отображение результата операции, из которого система может быть выведена действиями пользователя и переведена в состояние Обработка действий пользователя.

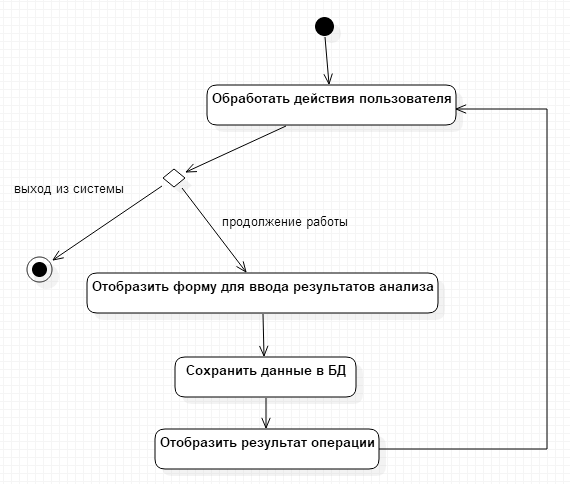
# Диаграммы деятельности

Диаграмма деятельности системы при авторизации в ней клиента мед-центра представлена ниже:



Первоначально система обрабатывает действие пользователя и выясняет, какого характера было это действие. При выходе клиента из системы, дальнейшая обработка действий завершается. При продолжении работы и запросе клиентом расписания, система защиты у базы данных расписание врачей. После чего, отобразив расписание, система снова обрабатывает действие пользователя. При продолжении работы и запросе клиентом личных данных, а именно своей карты, система запрашивает у базы данных карту клиента. После чего отображает её и снова обрабатывает действие клиента. При продолжении работы с системой и запросе клиентом личных данных, а именно списка предстоящих записей, система запрашивает предстоящие записи клиента у базы данных. После этого система отображает записи и далее обрабатывает действия пользователя.

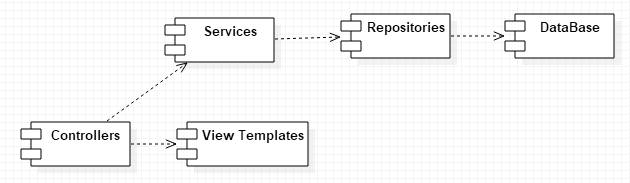
Диаграмма деятельности системы при авторизации в ней сотрудника лаборатории представлена ниже:



Первоначально система обрабатывает действие пользователя и выясняет, какого характера было это действие. При выходе клиента из системы, дальнейшая обработка действий завершается. При продолжении работы система отображает форму для ввода результата анализа. После ввода сотрудником лаборатории результатов и подтверждения, система сохраняет введенные данные в базу данных, отображает пользователю результат проведенной операции, после чего снова обрабатывает действия пользователя.

# Диаграмма компонентов

Диаграмма компонентов системы представлена на рисунке ниже:

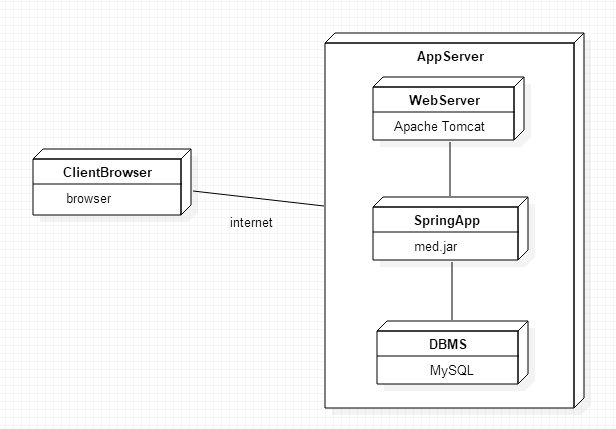


Система разделена на следующие компоненты:

* DataBase - компонент, представляющий API для работы с конкретной базой данных
* Repositories - компонент, представляющий собой группу классов, предоставляющих API для работы с базой данных путем абстрагирования от конкретной используемой базы данных и работающих с сущностями базы данных, и поэтому нуждающийся в компоненте DataBase
* Services - компонент, классы которого предоставляют API для использования определенных сервисов для работы с данными, использующий API классов компонента Repositories
* View Templates - компонент, представляющий собой набор документов, отвечающих за отображение страниц
* Controllers - компонент, являющийся связающим завеном между данными и их отображением. Он представляет API для взаимодействия со всем приложением.

# Диаграмма Развертывания

Диаграмма развертывания системы представлена на рисунке ниже:



Данную диаграмму развертывания можно поделить на две части:

* Клиентскую (представлена браузером клиента ClientBrowser).
* Серверную. Данная часть также разделена на три компонента:
  + WebServer (в данном случае сервер Apache Tomcat, принимающий запросы и передающий их приложению)
  + SpringApp (приложение, написанное, используя фреймворк Spring Web MVC, на языке Java)
  + DBMS (в данном случае используется MySQL)