|  |  |  |
| --- | --- | --- |
| **КУРСОВОЙ ПРОЕКТ** | | |
| МЕТОДЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ | | |
| по дисциплине «Архитектура программных систем» | | |
| Выполнил | | |
| студент гр. 5130904/20104 |  | Ганиуллин Р.Р. |
| Руководитель | | |
|  |  | Гончаров А.В. |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |

**Оглавление**

[Введение 2](#__RefHeading___Toc476_2553044399)

[Исходные данные 3](#__RefHeading___Toc474_2553044399)

[Маппинг 4](#__RefHeading___Toc472_2553044399)

[Архитектура приложения 5](#__RefHeading___Toc470_2553044399)

[Диаграмма классов 5](#__RefHeading___Toc478_2553044399)

[Сиквенс диаграмма 7](#__RefHeading___Toc482_2553044399)

[BPMN диаграмма 8](#__RefHeading___Toc480_2553044399)

[Модульная структура 9](#__RefHeading___Toc468_2553044399)

[Описание работы программы 9](#__RefHeading___Toc466_2553044399)

[Результаты работы и анализ 11](#__RefHeading___Toc464_2553044399)

[Вывод 11](#__RefHeading___Toc462_2553044399)

# Введение

Целью курсового проекта является разработка модели вычислительной системы (ВС) или отдельных её компонентов, обеспечивающей приближённое описание структуры и функциональности объекта на определённом уровне детализации. В основу модели закладывается возможность имитации поведения реальной системы для достижения требуемой точности и достоверности результатов.

Любая реальная вычислительная система обладает высокой сложностью, которая обусловлена множеством возможных состояний, внутренними и внешними связями, а также разнообразием анализируемых характеристик. Моделирование позволяет упростить изучение таких систем, сохраняя ключевые аспекты их поведения. Уровень детализации и приближения модели определяется поставленной задачей и требованиями к её реализации.

Существует несколько подходов к моделированию ВС, включая аналитические, аналоговые, физические и имитационные модели. В рамках данной работы применяется метод имитационного моделирования, который позволяет создать динамическую и гибкую модель, способную учитывать случайные процессы и многовариантность состояний. Одним из наиболее эффективных способов реализации таких моделей является представление системы в виде системы массового обслуживания (СМО).

Моделирование в формате СМО предоставляет инструменты для анализа процессов, связанных с обработкой заявок, управлением ресурсами и взаимодействием различных компонентов. Такой подход особенно полезен для оценки производительности, анализа задержек и оптимизации структуры системы. Курсовой проект направлен на демонстрацию возможностей метода имитационного моделирования при решении задач, связанных с проектированием и исследованием вычислительных систем.

# Исходные данные

Вариант 20. ИБ ИЗ2 ПЗ1 Д10З2 Д10О4 Д2П2 Д2Б1 ОР1 ОД2

Источники.

**ИБ** — бесконечный;

**И32**- равномерный

Приборы.

**ПЗ1** — экспоненциальный закон распределения времени обслуживания;

Дисциплины буферизации.

**Д1ОЗ3** — заполнение буферной памяти «на свободное место»;

Дисциплины отказа.

**Д1ОО4** — оследняя поступившая в буфер. Самая последняя заявка из поступивших в буфер, т. е. заявка, меньше других простоявшая в очереди, выбивается из БП, и на её место встаёт пришедшая заявка.

Дисциплины выбора заявок на обслуживание.

**Д2Б1** — FIFO (первым пришел — первым обслужен). Заявка, дольше всех простоявшая в буфере, будет выбрана на обслуживание раньше других.

Дисциплины выбора прибора.

**Д2П2** — по кольцу

Динамическое отражение результатов (пошаговый режим).

**ОД2** — формализованная схема модели, текущее состояние

Отражение результатов после сбора статистики ОР1-ОР2 (автоматический режим).

**ОР1** — отображение результатов в виде сводной таблицы результатов;

# Маппинг

Система моделирует работу поликлиники:

1. Источник - регистратура

2. Заявка — пациент с талоном

3. Буфер — скамейка (очередь пациентов)

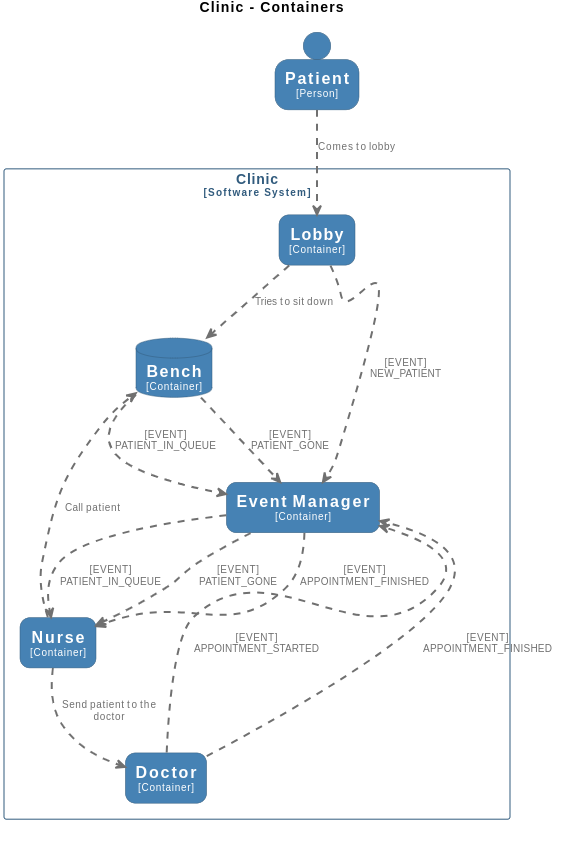
4. Диспетчер постановки — совмещен с источником (регистратура)

5. Диспетчер выборки - медсестра

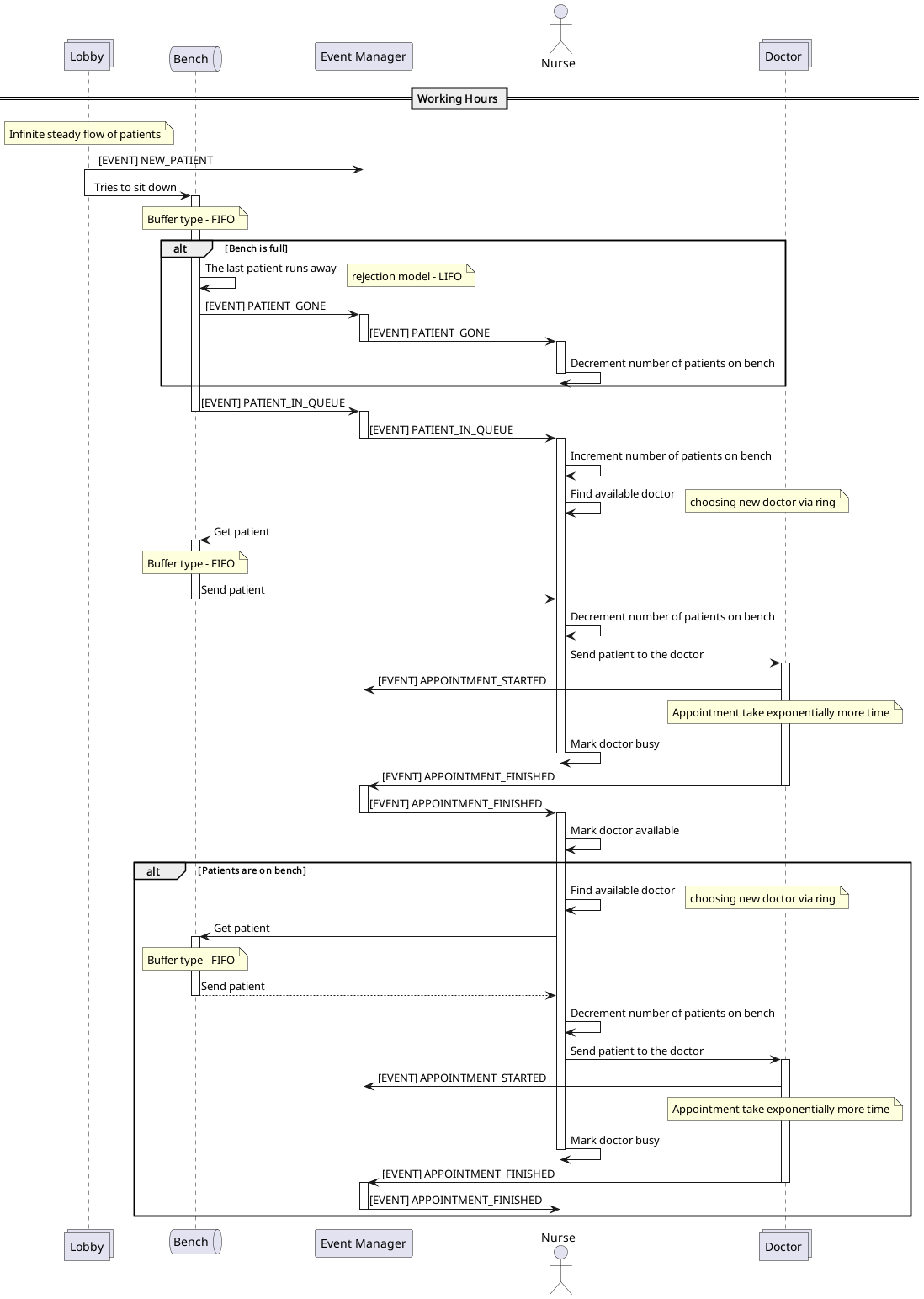
6. Прибор - врач

# Архитектура приложения

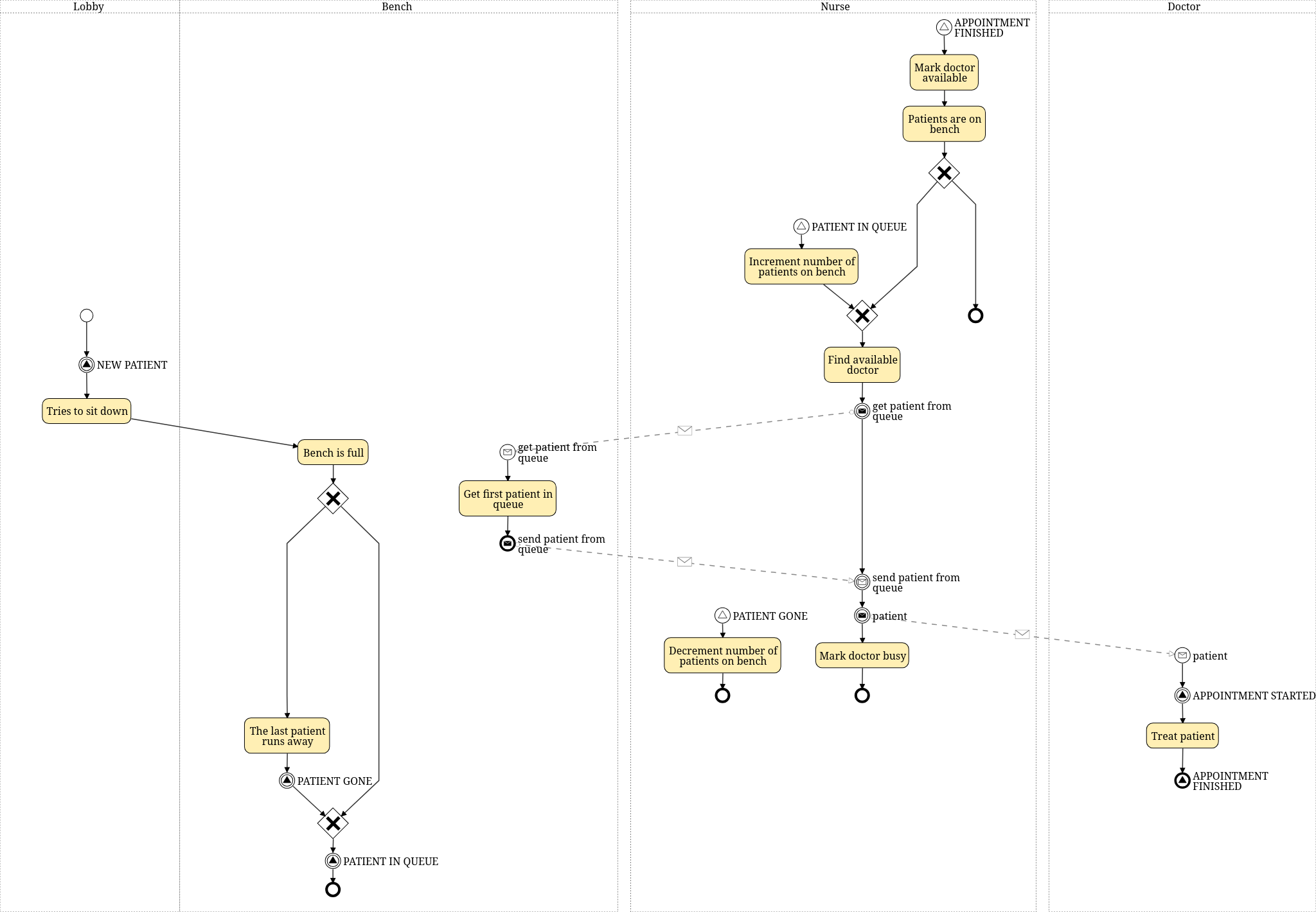
## Диаграмма классов



## Сиквенс диаграмма



## BPMN диаграмма



# Модульная структура

Программа написана на языке Go.

Приложение является объектно-ориентированным и состоит из следующих классов:

Lobby

* Реализует функциональность генерации заявок и постановку в буфер.

Patient

* Содержит всю необходимую информацию о пациенте.

Bench

* Управляет заявками в буфере. Осуществляет добавление, удаление и выбор заявок в соответствии с заданной дисциплиной обработки.

Doctor

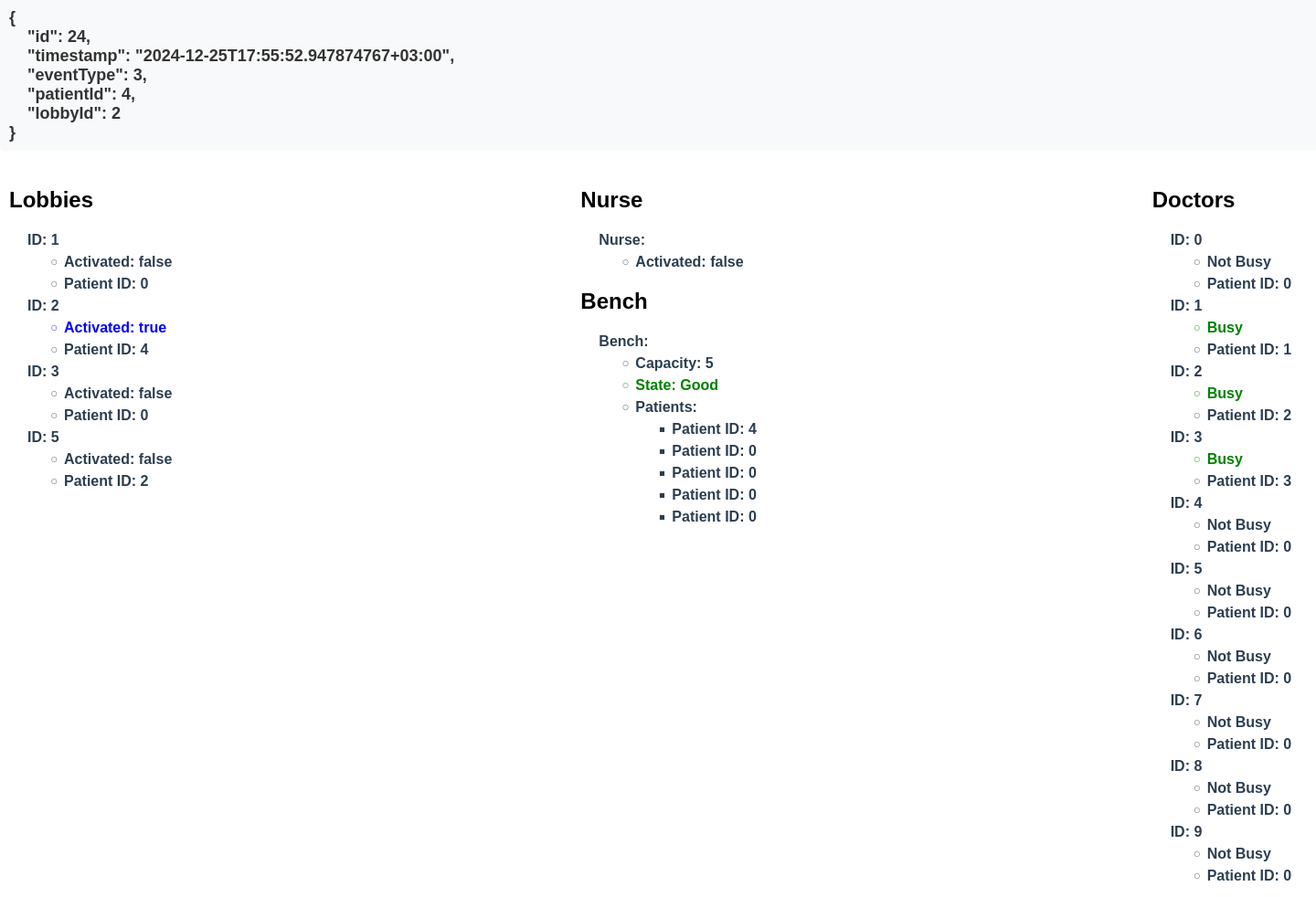
* Реализует функциональность обработки заявок

Nurse

* Отвечает за выбор заявок из буфера и их распределение между доступными врачами.

# Описание работы программы

Пошаговый режим работы отображает календарь событий, состояние буфера и текущую занятость специалистов



Автоматический режим работы выводит сводные таблицы

# 

# Результаты работы и анализ

Введем условные стоимости компонентов:

Врач – 30 000 у. е.

Одно место в буфере – 10 000 у. е.

За каждую отклоненную – 500 у. е.

За каждую единицу времени нахождения заявки в системе – 20 у. е.

Количество специалистов не может быть больше 13, количество мест в буфере не превышает 13.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Регистратура | Врач | Скамейка | Загруженность | P отк | Время в системе | Итог |
| 3 | 5 | 5 | 0,94 | 0,10 | 24 | 2354170 |
| 3 | 4 | 400 | 0,98 | 0,09 | 949 | 57370970 |
| 3 | 3 | 800 | 0,98 | 0.11 | 1245 | 78854890 |
| 30 | 45 | 200 | 0.99 | 0.13 | 82 | 75896110 |
| 30 | 50 | 50 | 0.99 | 0,09 | 17 | 28333620 |
| 30 | 55 | 10 | 0.96 | 0.04 | 19 | 20353460 |
| 60 | 60 | 60 | 0.93 | 0.18 | 33 | 33765420 |
| 60 | 70 | 50 | 0.91 | 0.11 | 30 | 32214720 |
| 60 | 80 | 20 | 0.85 | 0.07 | 24 | 26710730 |

Конфигурация, которая соответствует всем требованиям и является самой оптимальной выделена серым цветом.

# Вывод

В ходе работы в соответствии с заданным вариантом была смоделирована система массового обслуживания, а также написана программа на языке Go, имитирующая работу системы. Результаты, полученные с помощью программы необходимы для анализа работы и выбора подходящей конфигурации системы.