Документация на проект „Медицинско досие на пациента“

Изготвен от: „Слави Крумов Асенов“ F104490

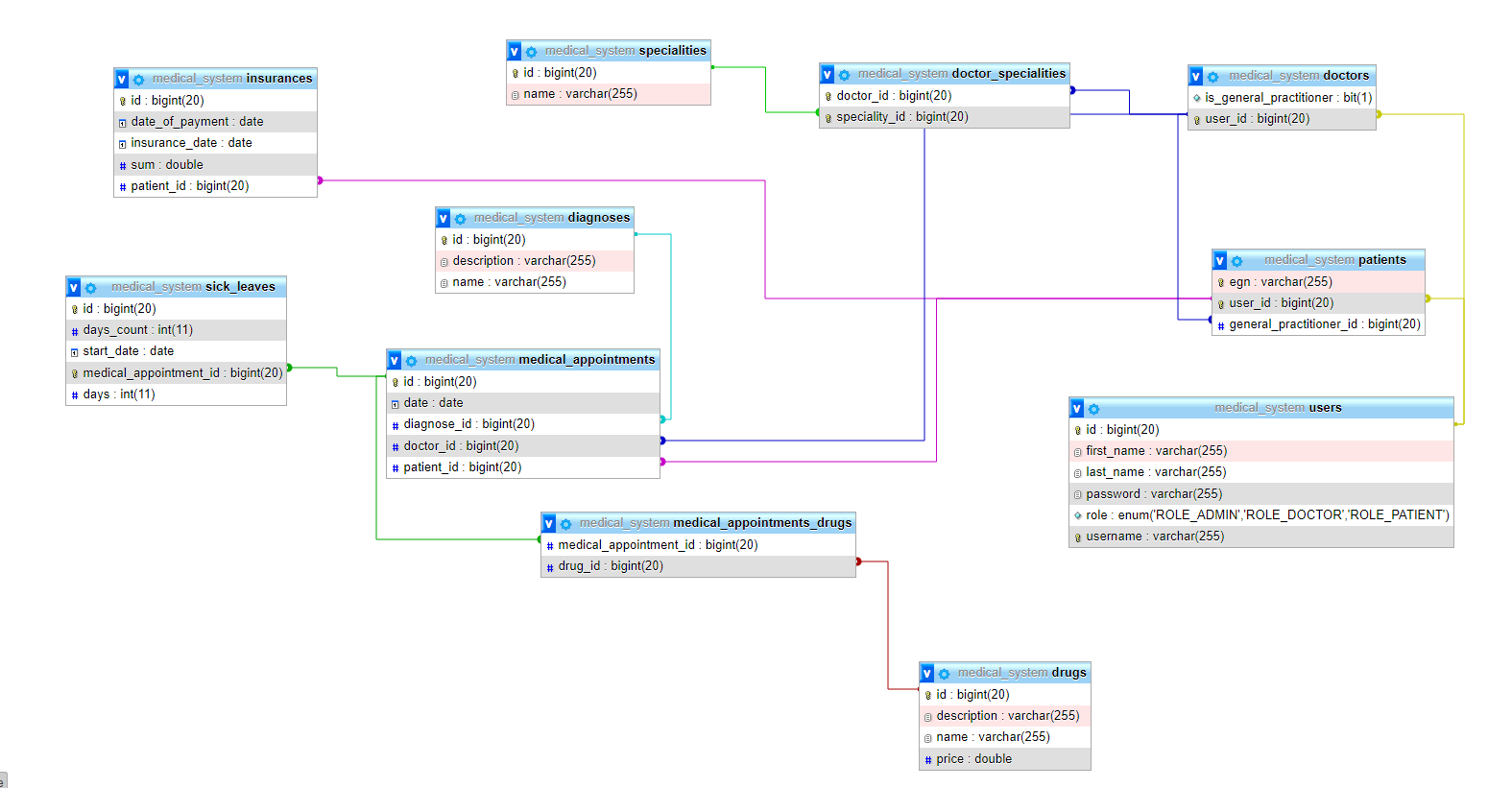
## Цел на документа

Да предостави информация за проекта, неговата структура, функционалности и главните проблеми и решенията им.

## Технически характеристики

Проектът е на Spring, като за БД е използвано MariaDB и Hibernate. Проектът НЕ е конзолно приложение, а е уеб-базирано, като за server-side Java templating engine е използвано Thymeleaf с **nz.net.ultraq.thymeleaf=>thymeleaf-layout-dialect**. Също така се за дизайна се ползва Bootstrap 5 за улеснение

## Структура на БД

[](db.jpg)

*(Снимката е линк към релативния път към снимката ctrl+mouse1)*

БД се състои от таблици за:

* Потребители (users)
* Доктори (doctors)
* Пациенти (patients)
* Специалности (specialities)
* Специалност на доктори (doctor\_specialities)
* Осигуровки (insurances)
* Диагнози (diagnoses)
* Лекарства (drugs)
* Прегледи (medical\_appointments)
* Лекарства разписани за прегледи (medical\_appointment\_drugs)
* Болнични (sick\_leaves)

## Логика в проекта

Когато се погледне структурата на БД, то изникват няколко неща:

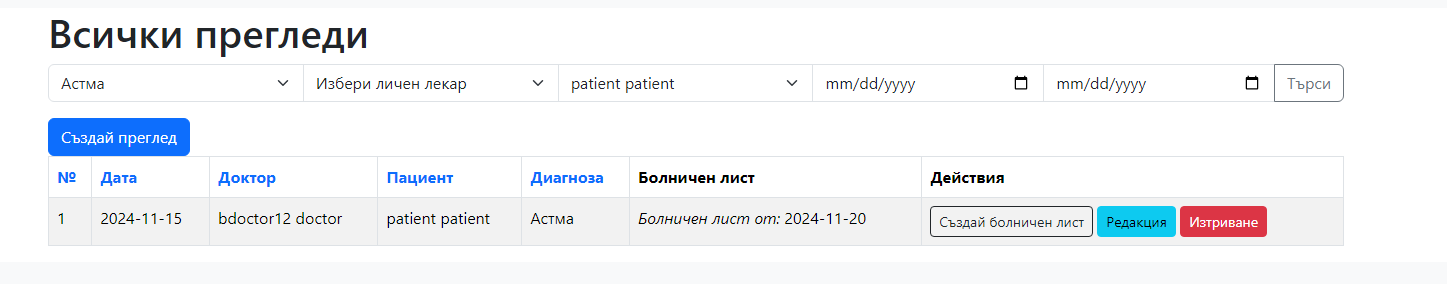
1. Логика по таблици.

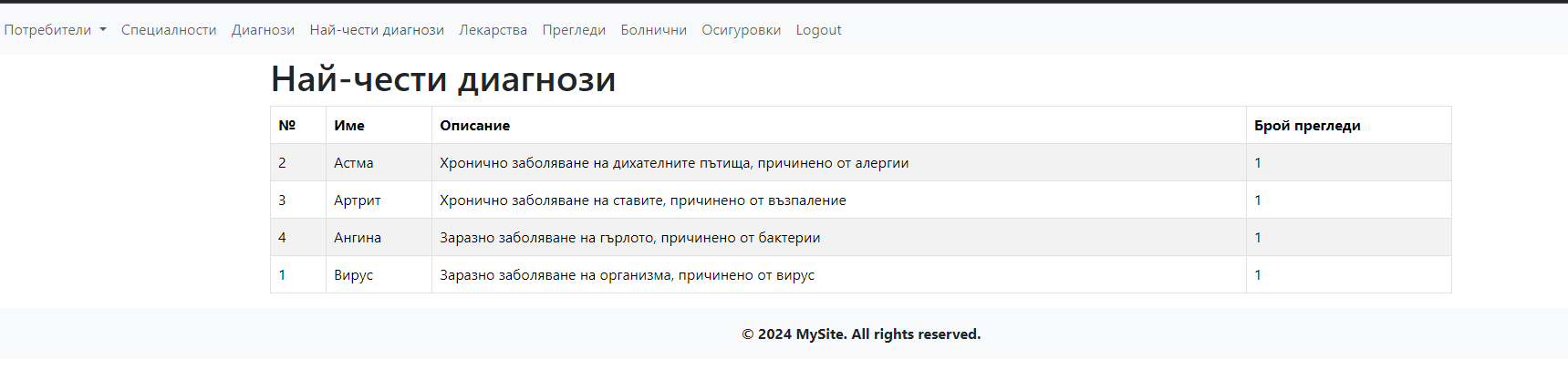
* Базова потребителска таблица (users) – тук се пази общата информация за пациенти, админи, лекари
* Доктори (doctors) – специфична информация за доктора
* Пациенти (patients) – специфична информация за пациента
* Специалности (specialities) – нищо особено, таблица с поле за името на специалността
* Специалност на доктори (doctor\_specialities) – M2M таблица
* Осигуровки (insurances) – тук се пазят плащанията на пациента за неговите осигуровки и от тук се гледа дали си е платил последните 6 месеца
* Диагнози (diagnoses) – нищо особено, таблица с поле за името и описанието на диагнозата
* Лекарства (drugs) – лекарства
* Прегледи (medical\_appointments) – връзка с пациент, диагноза, лекар; M2M
* Лекарства разписани за прегледи (medical\_appointment\_drugs) – M2M
* Болнични (sick\_leaves) – O2O с преглед

Бизнес слоя има най-различни валидации: уникалност, валидни стойности, минимални стойности, непразни и т.н. За валидация върху самите entity-та е ползван **spring-boot-starter-validation.** Всички те са вързани към формите и техните грешки. По този начин се показват грешките за всяко поле/правило. Валидацията е както на entity level, така и на DTO request обект level. Голяма част от полетата в таблиците са сортируеми, също така пагинацията, сортирането и филтрирането работят заедно. Направено е да може да не се губи подредбата при сменяне на страница, да не се рестартират данните на формата и т.н. По динамичните търсения където са в UI са направени с JPA Specifications.

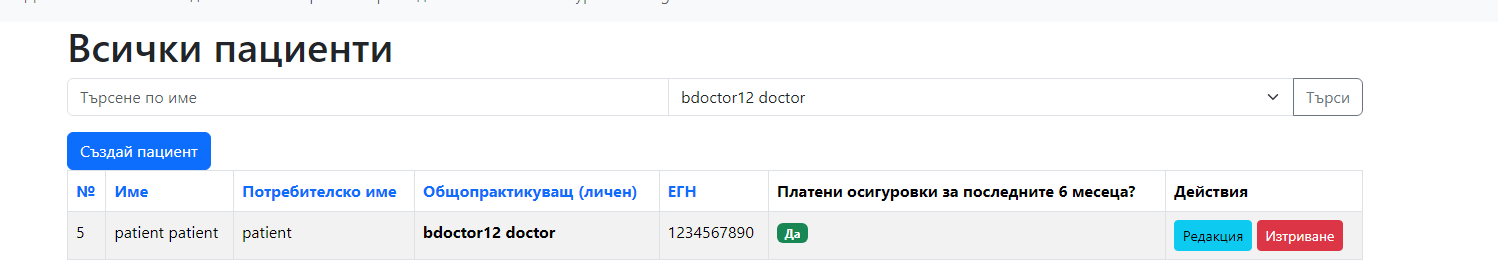
Справките, които бяха дадени в самото задание, се намират в самите страници на желания обект (посещения/прегледи, най-много болнични и т.н) са в съответните страници със съответните филтри:

Списък с пациенти, с дадена диагноза. – в прегледи; <http://localhost:8080/medical_appointments>

* 
* Информация за това, коя е диагнозата/диагнозите, които са диагностицирани
* най-често - <http://localhost:8080/diagnoses/most_seen>

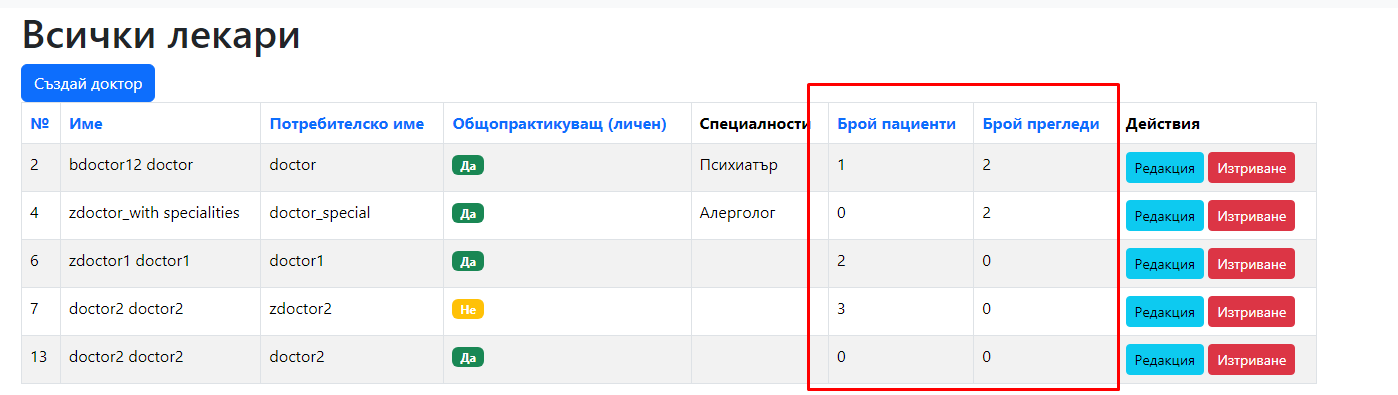


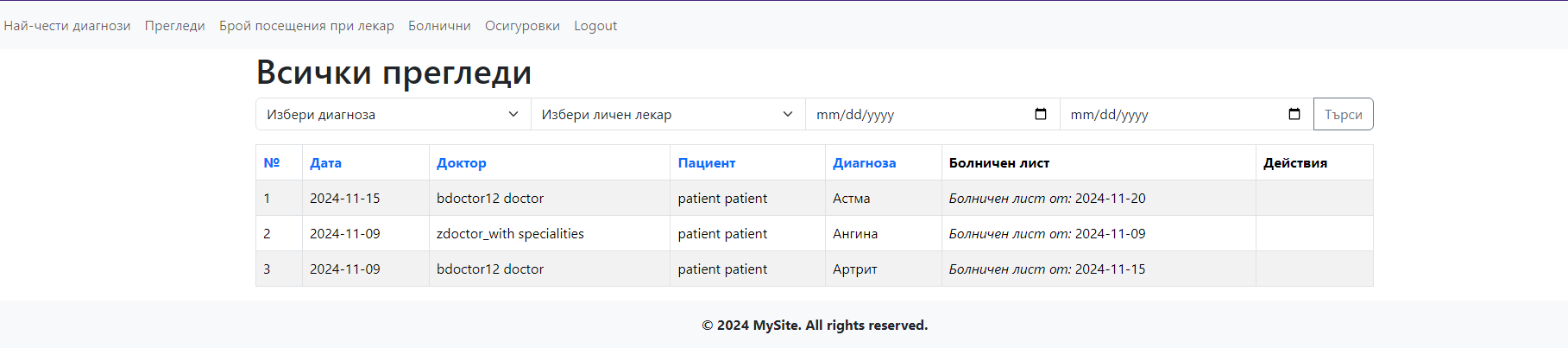
* Списък с пациенти, които имат даден личен лекар – пациенти - <http://localhost:8080/patients> gp dropdown

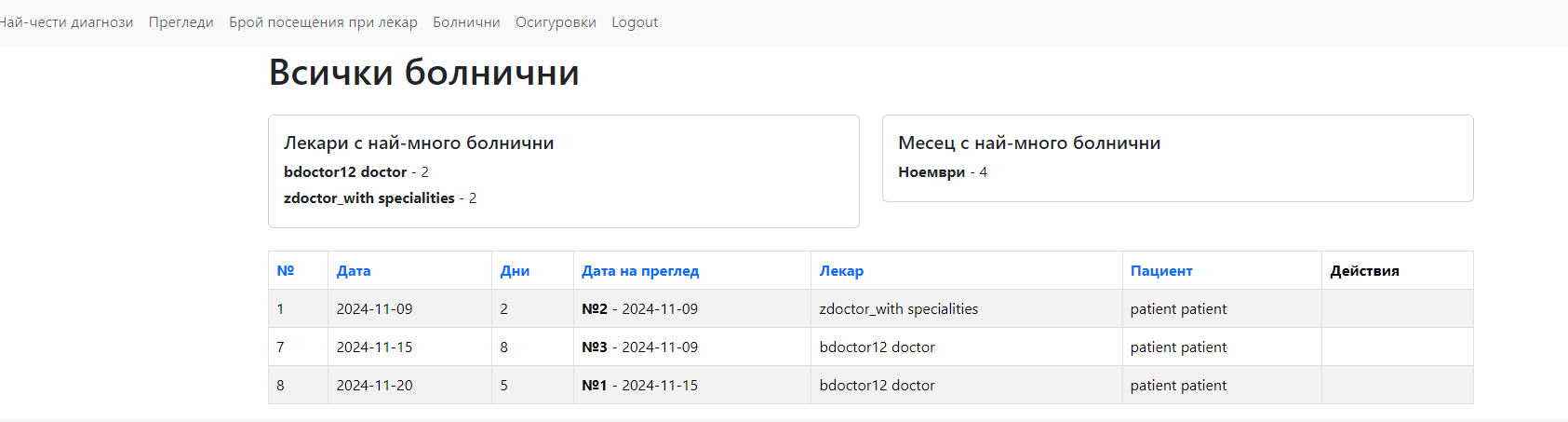


Брой на пациентите, записани при всеки от личните лекари в системата – в списъка с доктори

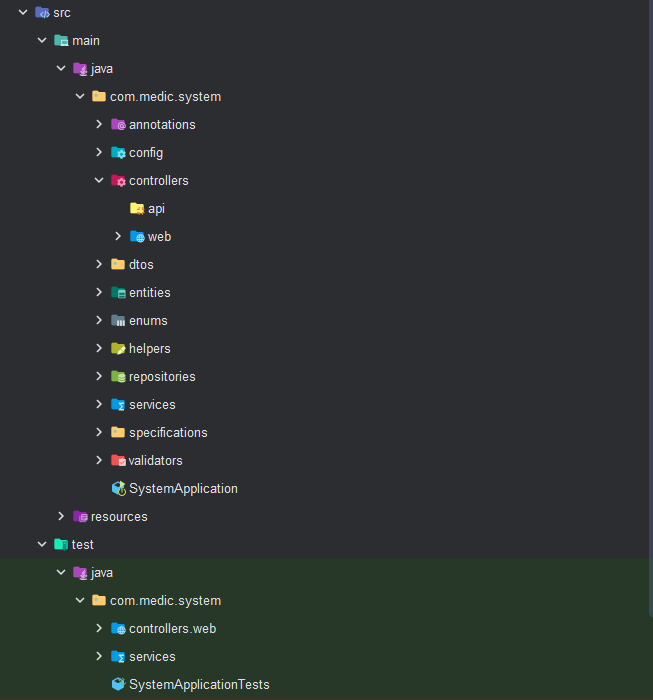
* Брой посещения при всеки от лекарите – в списъка с доктори



* Списък с посещения на всеки пациент – същото view за прегледи с няколко sec:authorize и различни query-та отзад
* Списък на прегледите при всички лекари в даден период – отново в прегледи с 2 полета за дата може да бъде постигнато
* Списък на прегледите при определен лекар за даден период – в прегледи с полетата за време и дропдауна за доктор
* Информация за месеца в годината, в който са издадени най-много болнични – списък с болнични – отгоре като статистика
* Информация за лекарят/лекарите, които са издали най-много болнични. списък с болнични – отгоре като статистика



## Структура на проекта и други характеристики



* **controllers** – папката с контролерите, тук са всички endpoint-и. Всеки контролер използва service-ите, които са нужни за дадените страници. Няма REST API контролери.
* **config –** тук се намират 3 файла: ThymeleafConfig за вкарването на 1 utils class в template-ите, security конфиг-а и DatabaseLoader-а, който 1 вид изпълнява функция на seeder.
* **annotations** – тук се намират 2 class level анотации използвани за валидация – FieldMatch за валидация дали 2 полета съвпадат и Unique за unique валидация на поле. И двете валидации трябва да са class level, за да имат достъп до нужните полета, тъй като не трябва да достъпвам само полето, което се валидира
* **validators –** тук са 2-та валидатор-а, които се ползват от анотациите
* **entities** – тук са всички класове за всяка 1 бд таблица
* **enums** – тук е избора за роля
* **dtos –** тук се намират обектите използвани във формите за създаване, редактиране и филтриране на различните обекти. Групирани по entity-то, с което работят
* **services** – тук е бизнес логиката. Всеки service, използва набор от repository класове, които са му нужни за логиката. Всеки service клас е анотиран с @Service.
* **helpers** – тук се намира 1 ThymeleafHelper, който ползвам за къстъм функции в template-тите
* **repositories** – тук се намират JPA repository-тата
* **specifications –** тук се намират JPA specification-ите, които се ползват в някои от service-ите за динамично сортиране и като цяло улесняват работата.
* **resources** –
  + static – тук са bootstrap minified файловете
  + templates – тук са thymeleaf темплейтите:
    - всеки темплейт се „наследява“ от layout.html
    - по специфичния файл тук е fragments/sorting.html, като това е всеки <th> елемент, който се сортира в List view-тата. И error темплейтите, които са направени да са по user friendly.

Проектът има и набор от тестове. Тук ще вметна, че главно тестовете са върху service-ите с Mockito. Има все пак и 1 controller тест

1. Заключение

Проектът „Медицинско досие на пациента“ изпълнява описаните функционални изисквания на приложението. То е уеб базирано с цел визуализация на данните през всеки 1 момент. Този документ описа основните характерности, правила и структура на проекта.