# Поликлиника

### 

[Поликлиника](#_iyrkzdx7a4jy)

[0. Изменения](#_tbmvnyj9c89k)

[1. Введение](#_1a6w12bcocrt)

[2. Описание системы.](#_4y2tcc1qjc3)

[2.1. Администраторы, врачи и пациенты.](#_fi8rksfwpnyv)

[2.2. Запись к врачу.](#_erheo3bu61x)

[3. REST-интерфейс](#_6jvau9gx1251)

[3.1. Общие правила](#_wy0g9gcjf5g1)

[3.2. Администратор - регистрация нового администратора.](#_do68cxn1jmgc)

[3.3. Администратор - регистрация врача и задание расписания приема.](#_j81f11ip22t4)

[3.4. Пациент - регистрация пациента.](#_bj8fiu449hrr)

[3.5. Login.](#_2ywm16oqmg2t)

[3.6. Logout.](#_h2ytru4qs5ac)

[3.7. Получение информации о текущем пользователе](#_k8vgnop2evz5)

[3.8. Получение информации о враче](#_v6ckfuv0yfl2)

[3.9. Получение информации о врачах](#_sdn6znj9f04)

[3.10. Администратор или врач - получение информации о пациенте](#_208srwfew65q)

[3.11. Администратор - редактирование профиля администратора](#_xqb0gdv2hhu1)

[3.12. Администратор - изменение расписания врача](#_u5d8p6fszkfl)

[3.13. Пациент -редактирование своего профиля](#_aklot67ll02c)

[3.14. Администратор - удаление врача](#_h3nbdbxfwpt8)

[3.15. Пациент - запись к врачу.](#_pte69jqv1z42)

[3.16. Отмена записи к врачу](#_km4kxb52ycvh)

[3.17. Врач - назначение врачебной комиссии.](#_ou23r5hl1cji)

[3.18. Пациент - отказ от приема на врачебной комиссии.](#_c4gexl6w6q0)

[3.19. Пациент - получение списка своих талонов на прием.](#_9bibhjf7tw6r)

[3.20. Получение статистики](#_kufgoldrhrc)

[3.21. Получение настроек сервера](#_yrr7vn3c6fkr)

[4. Отладочный API системы](#_6hcay57bkavh)

[4.1. Очистка состояния сервера](#_7zw3wjexjx67)

[5. Конфигурирование сервера.](#_omkzmc48xph)

[5.1. Конфигурационный файл.](#_sea3mis619k2)

[6. Требования по оформлению.](#_ds0fczdmztrh)

### 

## 0. Изменения

## 

|  |  |  |
| --- | --- | --- |
| № | Дата | Содержание |
| 1 | 6.03 | Уточнено имя artifactId в pom.xml. Должно быть net.thumbtack.school |
| 2 | 12.03 | В 3.12 исправлен Request в части задания расписания. Установлено то же, что и в 3.3 |
| 3 | 28.03 | Уточнен ряд URL’ов |
| 4 | 4.04 | Уточнены Response для 3.2 - 3.5. При регистрации администратора и врача cookie не возвращается, так как их регистрирует другой администратор. При регистрации пациента cookie возвращается, так как пациент регистрирует себя сам и после этого автоматически выполняется операция “логин”. При выполнении операции “логин” cookie возвращается всегда. |
| 5 | 10.04 | Уточнен Request в 3.14 |
| 6 | 13.04 | Уточнен формат даты и времени и дней недели (см. 1) |
| 7 | 18.04 | Уточнен URL в 3.2 |
| 8 | 25.04 | Исправлена ошибка в слове commission в URL в 3.17 и 3.18 |
| 8 | 6.05 | Уточнение по комиссии. Если у какого-то врача во время, отведенное на комиссию, нет приемов, комиссию назначить все равно можно - врач будет вызван для участия в комиссии |

### 

### 

## 1. Введение

Поликлинике нужен сайт, на котором администраторы поликлиники могли бы добавлять и удалять врачей, вести расписание приема врачей, а пациенты - записываться на прием к тому или иному врачу.

Вам предстоит написать прототип серверной части приложения, который будет работать 2 месяца с момента его ввода в пробную эксплуатацию.

Руководство поликлиники предполагает, что высказанные им требования к сайту являются логичными и непротиворечивыми. Так ли это в действительности или нет - должно выясниться в процессе разработки и пробной эксплуатации сайта. Если в ходе разработки или пробной эксплуатации появится необходимость корректировки тех или иных пунктов задания, то этот вопрос должен обсуждаться совместно представителями заказчика и разработчика с внесением в случае необходимости изменений в техническое задание. Изменения возможны как на клиентской, так и на серверной стороне.

Везде на сервере используется локальное (местное) время.

Дата всегда задается “дд-мм-гггг”, время - “чч:мм”. Дни недели указываются в виде текстовой строки - “Mon”, “Tue”, “Wed”, “Thu”, “Fri”

В выходные дни (суббота и воскресенье) поликлиника не работает. В данной версии программы не учитывается наличие праздничных дней.

## 2. Описание системы.

### 2.1. Администраторы, врачи и пациенты.

Администраторы и врачи на сервере самостоятельно не регистрируются, их регистрирует администратор. Изначально на сервере существует один администратор с именем admin и паролем, который задается при развертывании сервера. Этот администратор может зарегистрировать нового администратора, который получает такие же права на сервере.

Любой администратор может зарегистрировать врача, указав медицинскую специальность, по которой он будет вести приём и расписание приема.

Пациенты регистрируются на сервере самостоятельно .

Администраторы и пациенты могут изменить свои регистрационные данные, но не могут изменять логин. Список врачебных специальностей (терапевт, хирург и т.д.) задается в таблице БД , которая не подлежит редактированию средствами API данного сервера. Каждый врач может принимать только по одной специальности, хотя может быть несколько врачей одной и той же специальности. Каждому врачу назначается свой кабинет для приема. Номера кабинетов задаются в таблице БД, которая также не подлежит редактированию средствами API данного сервера. Номера кабинетов могут содержать нецифровые символы.

Администратор может изменить расписание приема врача, но только при условии, что не имеется записей к этому врачу, начиная с даты изменения, и на все последующие дни.

Администратор может удалить врача при увольнении последнего. В этом случае отменяются все записи на прием к нему, начиная с даты увольнения.

Администратор, задавая расписание приема, должен указать время, отводимое на одного пациента и график приема врача. На основании этих данных формируется ежедневное расписание приема врача, представляющее собой набор из дат и значений времени начала очередного приема. . Установленное администратором расписание действует вплоть до его изменения.

Врач может узнать свое ежедневное расписание приема, с указанием, какое время занято (в этом случае возвращаются также данные пациента) и какое свободно.

### 2.2. Запись к врачу.

Пациент может узнать расписание работы любого врача или всех врачей , или всех врачей указанной специальности, с указанием, какое время у врача/врачей свободно, и какое занято.

Пациент может записаться на прием (получить талон). При записи на прием пациент должен указать конкретного врача или специальность врача, к которому он записывается, дату и время. Если это время у врача свободно, производится запись на прием и выдается талон, в противном случае пациент получает отказ в записи. Записаться на прием можно только на дату, отстоящую от текущей не более, чем на 2 месяца.

После записи к врачу пациенту высылаются письмо на его e-mail и смс на его сотовый телефон[[1]](#footnote-0).

В сложных случаях может оказаться, что для лечения пациента необходимо собрать врачебную комиссию из нескольких врачей. Такую комиссию может назначить только врач, он же и выбирает время для комиссии - в это время все врачи, участвующие в комиссии, должны быть свободны, а пациент не должен быть записан к другому врачу или на другую комиссию.

Пациент может записаться к нескольким врачам, но не может записаться более одного раза к одному и тому же врачу на один и тот же день. Пациент не может записаться на время, которое даже частично пересекается с временем его другой записи на прием, в том числе и на врачебной комиссии.

Пациент может в любой момент до приема отказаться от талона, в этом случае указанное в талоне время приема врача вновь объявляется свободным. Пациент может отказаться от приема на врачебной комиссии.

Если врач был удален, всем записавшимся пациентам высылаются письма и смс об отмене приема. Все запланированные врачебные комиссии с участием этого врача отменяются, пациентам также высылаются письма и смс об отмене комиссии.

## 3. REST-интерфейс

### 3.1. Общие правила

Для взаимодействия клиентской и серверной частей приложения используется REST-интерфейс. Клиент посылает REST - запрос (Request), передавая в нем строку в формате JSON. Некоторые параметры передаются через заголовок HTTP-запроса. В ответ (Response) сервер возвращает свою строку, также в формате JSON, которая содержит информацию о результате выполненной операции.

Для идентификации администратора, врача и пациента используются cookie. При выполнении операции “Login” сервер возвращает cookie с именем “JAVASESSIONID”. Во всех остальных запросах предполагается, что клиент передает эту cookie.

Далее в документе принята следующая система обозначений.

Текст, содержащий латинские буквы и цифры, должен присутствовать в Request или Response в точности так, как написано. Текст, содержащий русские буквы, должен быть заменен на некоторое допустимое значение.

Например, "name":"имя пользователя" означает, что имя параметра должно быть “name”, а вместо “имя пользователя” должно быть указано некоторое допустимое имя.

Если значение параметра приведено в кавычках, параметр является текстовым, если без кавычек - целый типа int или long.

Порядок следования полей в Request и Response может быть произвольным.

При успешном выполнении всегда возвращается код HTTP 200.

В случае, если запрос не был выполнен из-за какой-то ошибки в нем, возвращается код HTTP 400 (для запроса по несуществующему URL - 404), и json следующего вида

{

"errors" : [  
 {  
 **"errorCode"**: "код ошибки",   
 **"field"**: "поле запроса, являющееся причиной ошибки",   
 **"message"**: "причина ошибки"  
 }

...  
]

}

Например

{

"errors" : [  
 {  
 **"errorCode"**: "LOGIN\_ALREADY\_EXISTS" ,  
 **"field"**: "login",   
 **"message"**: "User Ivanov already exists"   
 }  
 ]

}

### 3.2. Администратор - регистрация нового администратора.

POST /api/admins

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | {  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"position"**: "должность",  **"login"**: "логин",   **"password"**: "пароль"  } |

Логин может содержать только латинские и русские буквы и цифры и не может быть пустым. Пароль может содержать любые символы и тоже не может быть пустым. Максимальная длина логина, пароля, фамилии , имени и отчества не более ***max\_name\_length[[2]](#footnote-1)*** символов. Минимальная длина пароля ***min\_password\_length*** символов. Логин должен храниться так, как он задан, но не является case-sensitive при дальнейшей работе. Например, если администратор регистрировался с логином “Иванов”, он может впоследствии заходить на сервер, используя логины “Иванов”, “иванов”, “иВаНоВ” и т.д. Пароль является case-sensitive.

Имя, фамилия и отчество администратора могут содержать только русские буквы , пробелы и знак “минус” (используемый как тире).

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"position"**: "должность"  **}** |

### 

### 3.3. Администратор - регистрация врача и задание расписания приема.

POST /api/doctors

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"speciality"**: "специальность",  **"room"**: ”номер кабинета”,  **"login"**: "логин",   **"password"**: "пароль",  **"dateStart"**: "дата, начиная с которой создается расписание",  **"dateEnd"**: "дата, до которой создается расписание",  **"weekSchedule"**:  **{**  **“timeStart”** :"время начала приема",  **“timeEnd”** : "время окончания приема",  **“weekDays”** :  **[**  день недели 1.. день недели N  **]**  **},**  **"weekDaysSchedule"**:  **[**  **daySchedule :{**  **“weekDay”** : “день недели”  **“timeStart”** :"время начала приема",  **“timeEnd”** : "время окончания приема",  **},**  **...**  **],**  **"duration"**: время на прием одного пациента в минутах  **}** |

Требования к логину, паролю врача - те же, что и для администратора.

График работы врача задается одним из двух способов, поэтому в запросе должен обязательно присутствовать либо раздел “weekSchedule”, либо раздел “weekDaysSchedule”, но не оба вместе. Если часы приема врача одни и те же для любого дня недели, используется “weekSchedule” , в противном случае “weekDaysSchedule”. В weekSchedule поле weekDays может отсутствовать, или быть пустым списком, или быть равным null. Это означает, что врач принимает ежедневно.

Кабинет должен быть свободен.

Расписание создается на дни, начиная с dateStart и заканчивая dateEnd включительно.

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"speciality"**: "специальность",  **"room"**: ”номер кабинета”,  **"schedule"**: **[**  **{**  **“date”** :"дата приема",  **“daySchedule”: [**  **{**  **time** : “время”,  **patient** :  **{**  **"patientId"**: идентификационный номер пациента, получившего талон на это время,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"email"**: "адрес",  **"address"**: "домашний адрес, одной строкой",  **"phone"**: "номер",   **}**  **}**  **...**  **]**  **}**  **...**  **]**  **}** |

Поле “patient” присутствует в Response только если на это время уже выдан талон, в противном случае это поле должно отсутствовать. В Response для 3.3 это поле всегда отсутствует.

Response должен выдаваться упорядоченным по date, а в пределах данной date - упорядоченным по time

### 3.4. Пациент - регистрация пациента.

POST /api/patients

Request

|  |  |
| --- | --- |
| json | {   **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"email"**: "адрес",  **"address"**: "домашний адрес, одной строкой",  **"phone"**: "номер",   **"login"**: "логин",  **"password"**: "пароль"  } |

Требования к логину, паролю пациента - те же, что и для администратора.

E-mail должен соответствовать требованиям, предъявляемым к формату e-mail.

Домашний адрес не должен быть пустым, другие требования к нему не предъявляются.

Допустимые телефонные номера - сотовые номера любых операторов России. Номер может начинаться как с “8”, так и с “+7”. Наличие в номере знаков “-” (дефис) ошибкой не является, но перед записью в БД все знаки “-” удаляются. Номера телефонов стационарной связи указывать не разрешается.

Для зарегистрировавшегося пациента автоматически выполняется операция “Login” (п.3.4)

Response

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| json | {  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"email"**: "адрес",  **"address"**: "домашний адрес, одной строкой",  **"phone"**: "номер"  } |

### 

### 3.5. Login.

POST /api/sessions

Request

|  |  |
| --- | --- |
| json | {  **"login"**: "логин",   **"password"**: "пароль" } |

Response

Тот же, что и для 3.2, 3.3 или 3.4 соответственно. Во всех случаях также возвращается

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 3.6. Logout.

DELETE /api/sessions

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

### 3.7. Получение информации о текущем пользователе

GET /api/account

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

Тот же, что и для 3.2, 3.3 или 3.4 соответственно.

### 

### 

### 3.8. Получение информации о враче

GET /api/doctors/идентификатор\_врача

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| RequestParam | **schedule =** “yes | no” |
| RequestParam | **startDate =** “дата” |
| RequestParam | **endDate =** “дата” |

### 

Response

То же, что и в 3.3.

Если указано schedule = “no” или параметр schedule отсутствует,, расписание врача не возвращается.

Параметры startDate и endDate необязательные и имеют смысл только если указано schedule = “yes”. Если startDate не указан - расписание выдается, начиная с текущей даты. Если endDate не указан - вплоть до даты, отстоящей от текущей на 2 месяца.

Если запрос делает пациент, все поля в “patient” должны отсутствовать, за исключением случая, когда “patient” - это пациент, делающий запрос.

### 3.9. Получение информации о врачах

GET /api/doctors

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| RequestParam | **schedule =** “yes | no” |
| RequestParam | **speciality =** “специальность” |
| RequestParam | **startDate =** “дата” |
| RequestParam | **endDate =** “дата” |

### 

Параметр “speciality” является необязательным. Если он указан, выдается информация о врачах только этой специальности.

Параметры “schedule”, “startDate” и “endDate” имеют тот же смысл, что и в 3.8

Response

Список из элементов , возвращаемых в 3.8.

### 

### 3.10. Администратор или врач - получение информации о пациенте

GET /api/patients/идентификатор\_пациента

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

То же, что и в 3.4

### 3.11. Администратор - редактирование профиля администратора

PUT /api/admins

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

|  |  |
| --- | --- |
| json | {  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"position"**: "должность",   **"oldPassword"**: "прежний пароль",  **"newPassword"**: "новый пароль"  } |

Response

Тот же, что и в 3.2

Администратор может редактировать только свой профиль.

### 3.12. Администратор - изменение расписания врача

PUT /api/doctors/идентификатор\_врача

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"dateStart"**: "дата, начиная с которой вносятся изменения в расписание",  **"dateEnd"**: "дата, до которой вносятся изменения в расписание",  **"weekSchedule"**:  **{**  **“timeStart”** :"время начала приема",  **“timeEnd”** : "время окончания приема",  **“weekDays”** :  **[**  день недели 1.. день недели N  **]**  **},**  **"weekDaysSchedule"**:  **[**  **daySchedule :{**  **“weekDay”** : “день недели”  **“timeStart”** :"время начала приема",  **“timeEnd”** : "время окончания приема",  **},**  **...**  **],**  **"duration"**: время на прием одного пациента в минутах  **}** |

Правила, касающиеся задания расписания, те же, что и для 3.3

Если на все дни в интервале dateStart - dateEnd расписание не было ранее задано, изменения добавляются к ранее созданному расписанию.

Если на какие-то дни в интервале dateStart - dateEnd расписание уже было задано ранее и на эти дни не назначен ни один прием, новое расписание заменяет собой старое.

В остальных случаях запрос отвергается целиком.

Response

То же, что и в 3.3

### 3.13. Пациент -редактирование своего профиля

PUT /api/patients

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

|  |  |
| --- | --- |
| json | {   **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"email"**: "адрес",  **"address"**: "домашний адрес, одной строкой",  **"phone"**: "номер",   **"oldPassword"**: "прежний пароль",  **"newPassword"**: "новый пароль"  } |

Response

Тот же, что и в 3.4

### 3.14. Администратор - удаление врача

DELETE /api/doctors/идентификатор\_врача

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"date"**: "дата увольнения",  **}** |

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

### 

### 3.15. Пациент - запись к врачу.

POST /api/tickets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| json | **{**  **"doctorId"**: идентификатор врача,   **"speciality"**: "специальность врача",  **"date"**: "дата",  **"time"**: "время",  **}** |

Response

|  |  |
| --- | --- |
| json | **{**  **"ticket"**: “номер талона”,  **"doctorId"**: идентификатор врача,  **"firstName"**: "имя врача",   **"lastName"**: "фамилия врача",   **"patronymic"**: "отчество врача",   **"speciality"**: "специальность врача",  **"room"**: ”номер кабинета”,  **"date"**: "дата",  **"time"**: "время",  **}** |

В запросе может быть указан либо параметр “doctorId”, либо параметр “speciality”, но не оба одновременно. Если указан параметр “doctorId” - требуется записать на прием к указанному врачу. Если указан параметр “speciality” - к любому врачу этой специальности, выбор врача производится случайным образом.

Номер талона формируется следующим образом

D<идентификатор\_врача>DDMMYYYYHHMM

где DDMMYYYY - дата приема , HHMM - время приема

### 3.16. Отмена записи к врачу

DELETE /api/tickets/номер\_талона

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

Отменить прием может пациент, записавшийся на прием, врач, к которому он записался , а также администратор.

### 3.17. Врач - назначение врачебной комиссии.

POST /api/commissions

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| json | **{**  **"patientId"**: идентификатор пациента,  **"doctorIds"**:  **[**  идентификатор врача1,...идентификатор врачаN  **],**  **"room"**: ”номер кабинета”,  **"date"**: "дата",  **"time"**: "время",  **"duration"**: время, отведенное на комиссию в минутах  **}** |

Response

|  |  |
| --- | --- |
| json | **{**  **"ticket"**: “номер талона на комиссию”,  **"patientId"**: идентификатор пациента,  **"doctorIds"**:  **[**  идентификатор врача1,...идентификатор врачаN  **]**  **"room"**: ”номер кабинета”,  **"date"**: "дата",  **"time"**: "время начала",  **"duration"**: время, отведенное на комиссию в минутах  **}** |

Врач, по чьей инициативе назначается врачебная комиссия, автоматически включается в нее, независимо от того, указан ли он в разделе doctorsIds или нет. В комиссию может входить любое количество врачей (>1).

Кабинет для проведения комиссии - кабинет одного из врачей - членов комиссии.

Для создания комиссии необходимо, чтобы в течение всего интервала времени, отведенного на нее, все врачи, участвующие в ней были свободны.

Если в результате назначения комиссии у того или иного врача образуется остаток интервала времени, отведенного на прием одного пациента до или после комиссии, этот остаток считается свободным у врача и запись пациентов на это время не производится. Например, если исходно у врача были запланированы приемы в 13.00, 13.15 и 13.30, каждый по 15 минут, а комиссия назначена с 13.10 продолжительностью в 30 минут, то у врача образуются 2 свободных интервала : с 13.00 до- 13.09 включительно и с 13.40 до 13.44 включительно.

Номер талона формируется следующим образом

СD<идентификатор\_врача1>D<идентификатор\_врача2>...DDMMYYYYHHMM

где DDMMYYYY - дата приема , HHMM - время приема

### 

### 3.18. Пациент - отказ от приема на врачебной комиссии.

DELETE /api/commissions/номер\_талона\_на\_комиссию

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{}** |

Если пациент отказывается от приема на комиссии, у всех врачей, участвующих в комиссии, восстанавливается их расписание на то время, которое было отведено на комиссию.

### 

### 

### 3.19. Пациент - получение списка своих талонов на прием.

GET /api/tickets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{**  **[**  **"ticket"**: “номер талона”,  **"room"**: ”номер кабинета”,  **"date"**: "дата",  **"time"**: "время",  // для случая приема у одного врача  **"doctorId"**: идентификатор врача,  **"firstName"**: "имя врача",   **"lastName"**: "фамилия врача",   **"patronymic"**: "отчество врача",   **"speciality"**: "специальность врача",  // для случая приема на врачебной комиссии  **[**  **{**  **"doctorId"**: идентификатор врача,  **"firstName"**: "имя врача",   **"lastName"**: "фамилия врача",   **"patronymic"**: "отчество врача",   **"speciality"**: "специальность врача",  **}**  **...**  **]**  **...**  **]**  **}** |

### 

### 3.20. Получение статистики

Этот эндпойнт Вам предстоит разработать самостоятельно. Должна выдаваться статистика по каждому врачу, по всем врачам поликлиники и по каждому пациенту за любой указанный в запросе период времени. Необходимо обеспечить гибкую настройку объема выдаваемых данных так, чтобы в зависимости от параметров запроса выдавалась бы или только суммарная, или подробная информация.

### 

### 

### 3.21. Получение настроек сервера

GET /api/settings

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Параметр “cookie” для этого запроса не является обязательным. Если он передается, то для администратора выдаются доступные ему настройки, для врача - доступные ему, а для пациента - доступные ему. Если cookie не передается в запросе, возвращается список настроек, доступных до выполнения операции “Login”

Response

В настоящее время для всех 4 случаев выдается один и тот же результат. Это поведение может быть в дальнейшем изменено.

|  |  |
| --- | --- |
| json | **{**  **"maxNameLength"**: значение ***max\_name\_length,***  **"minPasswordLength"**: значение ***min\_password\_length***  **}** |

## **4. Отладочный API системы**

Отладочный API предназначен для управления сервером со стороны клиента во время отладки и тестирования системы. В рабочей версии продукта этот АПИ должен быть удален или деактивирован, поскольку представляет угрозу для безопасности. Поэтому рекомендуется эти методы вынести в отдельный класс/файл так, чтобы их можно было легко удалить.

Все команды отладочного API выполняются без указания пользователя, поэтому cookie в них не передается и проверка прав не производится. Команды всегда возвращают успешный ответ.

В настоящее время имеется только один отладочный метод. Разработчик вправе добавить иные отладочные методы, если посчитает необходимым.

### 

### 

### 

### 

### **4.1. Очистка состояния сервера**

POST /api/debug/clear

Request

Отсутствует

Response

|  |  |
| --- | --- |
| json | {} |

### 

Удаляет с сервера все данные, кроме учетной записи admin. Метод используется в тестах, чтобы очередной тест не зависел от состояния сервера, получившегося в результате выполнения предыдущего теста. Метод предназначен для отладки, в production должен быть отключен.

### 

## **5. Конфигурирование сервера.**

### **5.1. Конфигурационный файл.**

Конфигурационный файл (application.properties) используется для задания параметров работы сервера и содержит элементы вида “параметр = значение”. Он должен находится в каталоге src\main\resources Каждый элемент находится в отдельной строке файла (таблица 1). Порядок следования элементов в файле произвольный. Все параметры обязательные. Формат данных в конфигурационном файле всегда правильный, проверять не требуется. В файле могут быть и другие параметры, например, параметры для настройки Spring.

Параметры конфигурационного файла недоступны пользователям. Во время работы сервера параметры конфигурационного файла изменяться не могут. Для того, чтобы измененные параметры вступили в действие, необходим перезапуск сервера. После перезапуска измененные значения параметров используются только для новых запросов, никакие изменения в БД для ранее выполненных запросов не производятся.

**Таблица 1. Параметры конфигурационного файла**

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Назначение |
| server.port | int | Порт, на котором работает REST-сервер. |
| max\_name\_length | int | Максимальная длина имени |
| min\_password\_length | int | Минимальная длина пароля |

Пример конфигурационного файла

server.port = 8888

max\_name\_length = 50

min\_password\_length = 10

## **6. Требования по оформлению.**

В pom.xml groupId и artifactId должны быть заданы следующим образом :

<groupId>net.thumbtack.school</groupId>

<artifactId>hospital</artifactId>

Класс, содержащий метод main, должен находиться в пакете net.thumbtack.school.hospital и иметь имя HospitalServer

Имена остальных пакетов и классов - на усмотрение разработчика, но рекомендуется следовать общепринятому способу именования, аналогично Заданию 11.

Имя БД - hospital. Имена таблиц - на усмотрение разработчика. Для имен таблиц рекомендуется использовать единственное число (например, “doctor”, а не “doctors”).

1. Реализация отсылки писем и смс в сервере в настоящее время не требуется, достаточно записать в лог, что письмо и смс были отправлены. [↑](#footnote-ref-0)
2. Здесь и далее слова, выделенные жирным курсивом, обозначают параметры конфигурационного файла [↑](#footnote-ref-1)