

Задания к курсу

“Разработка прикладных компьютерных систем”

3 курс, 2021-2022 уч. г. (Часть 3)

8. Реализуйте функционал для игры в фризские шашки (вариант 5x5 - Frysk!). Необходимые контракты (интерфейсы) и их реализации (фигуры, доска, ходы, нотация и т. д.) в соответствии с правилами, продумайте самостоятельно. О правилах игры можно прочитать здесь:
- <https://lidraughts.org/variant/standard>
<https://lidraughts.org/variant/frisian>
<https://lidraughts.org/variant/frysk>

9. Реализуйте асинхронный сервер-посредник для отправки запросов в базу данных, хранящую информацию об игроках. База данных должна содержать следующие данные:

- логины и “солёные” логином хэши паролей (хэш-функцию выберите самостоятельно) игроков с индексированием по логину;
- [опционально] информацию для восстановления пароля (контрольный вопрос (один из предложенных или свой, хэш ответа на вопрос));
- [опционально] информацию об игроках: дату регистрации, общее время активности на сервере с момента регистрации, дату последней активности;
- [опционально] информацию о блокировке со стороны администрации игрока (срочной и бессрочной) с указанием причины.

Схему базы данных определите самостоятельно. Все таблицы в базе данных должны находиться в третьей+ нормальной форме. Структуру запросов к сервису определите самостоятельно (см. другие задания). Информацию о подключении к базе данных получайте из файла конфигураций (предусмотрите трансформации для конфигураций).

10. На базе технологии gRPC реализуйте асинхронный централизованный сервер, обслуживающий партии игроков в фризские шашки. Зависимые сервисы получайте через конструктор из выбранной Вами реализации DI/IoC (Вы можете использовать DryIoC, IServiceProvider, Autofac, Windsor Castle, Unity, etc.). Предусмотрите следующие запросы:

- авторизация игрока по логину и паролю (запрещено передавать пароль в открытом виде) ([опционально] с уведомлением о блокировке со стороны администрации);
- создание новой игры:

- функционал по умолчанию: авторизовавшись, игрок ждёт другого игрока (либо присоединяется к уже ожидающему игроку), после чего начинается новая партия;
- [опционально] реализуйте функционал игрового лобби (с отображением лобби и взаимодействием с ним на стороне клиента);
- выполнение игроком хода с его сохранением на сервере в историю ходов партии и с уведомлением о ходе клиентского приложения соперника с указанием дополнительной информации (например, игра завершена);
- сдача игроком партии по его запросу с уведомлением об этом клиентского приложения соперника;
- предложение игроком ничьей по его запросу с уведомлением об этом клиентского приложения соперника;
- принятие/непринятие игроком предложенной соперником ничьей с уведомлением об этом клиентского приложения соперника и с завершением при этом партии;
- [опционально] отмена партии (возможна только до выполнения игроками первых двух полуходов - обеспечьте проверку);
- [опционально] отключение игрока посредством завершения его клиентского приложения с автоматическим поражением и уведомлением соперника о победе;
- [опционально] отправка текстового сообщения в игровой чат партии;
- [опционально] ведите лог-файл с указанием подробной информации о событиях, происходящих во время работы сервера (логи должны быть сгруппированы в разные файлы по дате записи лога).

Уведомление клиентов производите через инстанс gRPC-клиента (nested to gRPC-сервер), полученного gRPC-сервером из DI-контейнера.

11. Реализуйте оконное приложение на WPF для асинхронного взаимодействия с централизованным сервером, реализованным в задании 10. Клиентское приложение должно предоставлять следующий функционал:

- регистрация нового игрока (пара логин/пароль, с обеспечением уникальности логина);
- авторизация игрока по логину/паролю ([опционально] с уведомлением о блокировке со стороны администрации);
- [опционально] восстановление пароля игрока посредством ответа на контрольный вопрос и указанием нового пароля;
- присоединение к игре:
 - функционал по умолчанию: присоединение к игре по нажатию кнопки;
 - [опционально] создание новой “комнаты” в лобби, либо присоединение к уже существующей “комнате” (отображайте

логины игроков в “комнате”); также предусмотрите функционал по отображению/скрытию заполненных в лобби “комнат”;

- во время игры:
 - [опционально] запрос на отмену партии посредством нажатия кнопки (кнопка визуализирована и доступна только до выполнения первых двух полуходов в партии);
 - выполнение игроком хода посредством взаимодействия игрока с графическим интерфейсом ([опционально] drag&drop);
 - [опционально] отображение возможных ходов в соответствии с правилами;
 - запрос на сдачу партии;
 - предложение сопернику ничьей;
 - принятие или непринятие предложенной соперником ничьей;
 - [опционально] элемент управления, отображающий полуходы игроков в партии (запись полуходов должна быть минимизирована и однозначно декодируема из позиции до выполнения конкретного полухода);
 - [опционально] возможность перехода на позицию, получившуюся после конкретного полухода при нажатии ЛКМ на соответствующее отображение модели полухода;
 - [опционально] элемент управления, отображающий текстовые сообщения из чата партии и позволяющий отправить в чат сообщение;
- [опционально] при наличии лобби, реализуйте общий чат игроков, ожидающих в лобби;
- [опционально ТОЛЬКО при наличии фичи drag&drop] Приложение должно быть реализовано на базе архитектуры MVVM;
- [опционально] ведите лог-файл с указанием подробной информации о событиях, происходящих во время работы клиентского приложения (логи должны быть сгруппированы в разные файлы по дате записи лога);
- объекты отображений (View), моделей отображений (ViewModel), сервисов, необходимых для клиентского приложения, должны быть получены из глобально доступного DI-контейнера;
- Предусмотрите отдельный сервис-посредник для отправки запросов по gRPC;
- Обработку запросов от централизованного сервера реализуйте посредством развёртывания на стороне клиента gRPC-сервера, общающегося с моделями отображений при помощи событий.

12. Доработайте функционал gRPC-сервера из задания 10 следующим образом:

- Реализуйте механизмы бинарных [де]сериализации для партий;
- Вынесите хранение активных партий из gRPC-сервера в Redis (ключ партии собирайте относительно логинов игроков) при помощи реализованных механизмов [де]сериализации; хранение информации о партиях в рамках инстанса сервера больше недопустимо;
- по мере протекания партий обновляйте их состояние в Redis;
- по завершении партии удалите её сериализованное состояние из Redis.

Для подключения к Redis реализуйте сервис-фабрику, создающую объекты подключений. Фабрика должна быть доступна из gRPC-сервера и должна быть получена им [сервером] из DI-контейнера.

13. На базе технологии gRPC реализуйте асинхронный сервер-посредник для отправки запросов в базу данных, хранящую информацию о сыгранных партиях. База данных должна содержать следующие данные:

- сериализованное состояние партии;
- информацию об игроках, сыгравших партию;
- дата и время начала/окончания игры.

Схему базы данных определите самостоятельно. Все таблицы в базе данных должны находиться в третьей+ нормальной форме. Структуру запросов к сервису определите самостоятельно (см. другие задания). Информацию о подключении к базе данных получайте из файла конфигураций (предусмотрите трансформации для конфигураций).

14. Доработайте функционал заданий 12 и 13 таким образом, чтобы после удаления сериализованного состояния завершенной партии из Redis, она была сохранена в базу данных сыгранных партий. Отправку запроса на сохранение партии реализуйте через брокер сообщений RabbitMQ. Информацию о подключении к брокеру сообщений получайте из файла конфигураций (предусмотрите трансформации для конфигураций).

15. Доработайте функционал оконного приложения WPF из задания 11, предоставив игроку возможность загрузки своих партий с возможностью их пошагового просмотра: предусмотрите кнопки undo/redo выполнения полуходов, возврата к началу партии, перехода к концу партии, перехода к произвольному месту в партии (при наличии элемента управления, отображающего полуходы в партии).

16. Реализуйте оконное приложение на WPF, предназначенное для взаимодействия с сервисом из задания 9.

17. [опционально] Дальнейшие возможные улучшения:

- кастомизация вариантов правил (русские, международные, канадские, турецкие, поддавки, столбовые, ...);
- игра на время (контроли времени на партию с добавлением времени на ход);
- редактор доски;
- тренажёры для решения задач;
- <придумать своё> :)