**Software Architecture Document (v 01.11.23)**

**1.** **Introduction (Введение)**

Представленная в данном документе информация определяет слои, на которые делится приложение, а также интерфейсы их взаимодействия. Благодаря этим данным возможно оценить сложность системы, возможности ее улучшения и принять меры по дальнейшей поддержке и обслуживанию/масштабированию в долгосрочной перспективе.

**1.1** **Purpose**

Данный документ содержит описание архитектуры программного обеспечения с целью формирования четкого представления структуры, компонентов и взаимодействий внутри разрабатываемой системы. Документ выступает в качестве образца, позволяющего визуализировать дизайн и организацию системы в целом. В качестве эффективного представления информации выступают UML-диаграммы.

**1.2 Scope (Область применения)**

Документ разрабатывается в рамках проекта “Финист” - системе знакомств для серьезных отношений в старославянском сеттинге на основе RUP методологии. Предназначен для заинтересованных лиц, лиц участвующих в разработке и других членов команды.

**1.3 Definitions, Acronyms and Abbreviations (Определения и аббревиатуры)**

Для дальнейшего понимания документа важно изучить роли пользователей системы, которые они принимают во время исполнения бизнес-процесса. Значение всех терминов дано в глоссарии проекта:

[Glossary](https://docs.google.com/document/d/1reBBvZCW8-QsYr5F5Ixum-bbKSkMxmGXP-yf2m_T3RU/edit)

**1.4 References (Ссылки)**

[SRS](https://docs.google.com/document/d/1KZPhcczRg4XHN3is2uTxyx07Gj8I-aY-/edit?usp=drive_link&ouid=108798662073330685189&rtpof=true&sd=true)

[UCS](https://docs.google.com/document/d/1ydDMM3psIyYvH4MX7Mlkt2e8n_XKm07K/edit?usp=drive_link&ouid=108798662073330685189&rtpof=true&sd=true)

**1.5 Overview (Обзор документа)**

**Вторая глава** описывает архитектуру разрабатываемой системы в целом.

**Третья глава** описывает цели и ограничения архитектуры системы.

**Четвертая глава** содержит описание основных сценариев использования системы разными типами пользователей.

**Пятая глава** содержит описание слоев, на которые делится приложение, а также интерфейсов их взаимодействия.

**Шестая глава** описывает систему как совокупность взаимодействующих процессов, привязанных к определенным моментам времени.

**Седьмая глава** содержит описание конфигурации файлов, из которых состоит система, мест их расположения и описание взаимодействия их друг с другом.

**Восьмая глава** содержит подробное описание системы в уже реализованном виде.

**Девятая глава** описывает основные характеристики измерения производительности системы и их границы, которые могут оказать влияние на архитектуру.

**Десятая глава** описывает, каким образом архитектура системы удовлетворяет ее показателям качества - масштабируемости, надежности, мобильности, безопасности и т.д.

**2.** **Architectural Representation (Представление архитектуры)**

| ***Diagram\View*** | ***Use Case View*** | ***Logical View*** | ***Implementation view*** | ***Process view\*\**** | ***Deployment View*** |
| --- | --- | --- | --- | --- | --- |
| ***Use Case Diagram*** | + |  |  |  |  |
| ***Class Diagram*** | + (Взаимодействие сущностей) | +  (Описание основных классов и интерфейсов их взаимодействия) | +  (Полное описание классов с указанием их методов/полей, указать типы связей между классами) |  |  |
| ***Activity Diagram*** | +  (Абстрактное описание) | +  (Более подробное описание, уровни взаимодействия) | +  (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов). |  |  |
| ***State Machine Diagram*** | +  (Абстрактное описание) | +  (Более подробное описание, уровни взаимодействия) | +  (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов). |  |  |
| ***Sequence Diagram*** | +  (Абстрактное описание) | +  (Более подробное описание, уровни взаимодействия) | +  (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов). |  |  |
| ***Cooperative Diagram*** | +  (Абстрактное описание) | +  (Более подробное описание, уровни взаимодействия) | +  (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов). |  |  |
| ***Package Diagram*** |  | + | - |  |  |
| ***Data Base Diagram*** |  |  | +  (Полная ER модель базы данных  + её даталогическая модель) |  |  |
| ***Deployment Diagram*** |  |  |  |  | +  (Подробная диаграмма развертывания с указанием характеристик машин и интерфейсов взаимодействия) |
| Timeline diagram |  |  |  | + |  |

**3.** **Architectural Goals and Constraints (Цели и ограничения архитектуры)**

Технические ограничения архитектуры прямо следуют из оборудования заказчика. ПО будет развёрнуто на сервере Helios:

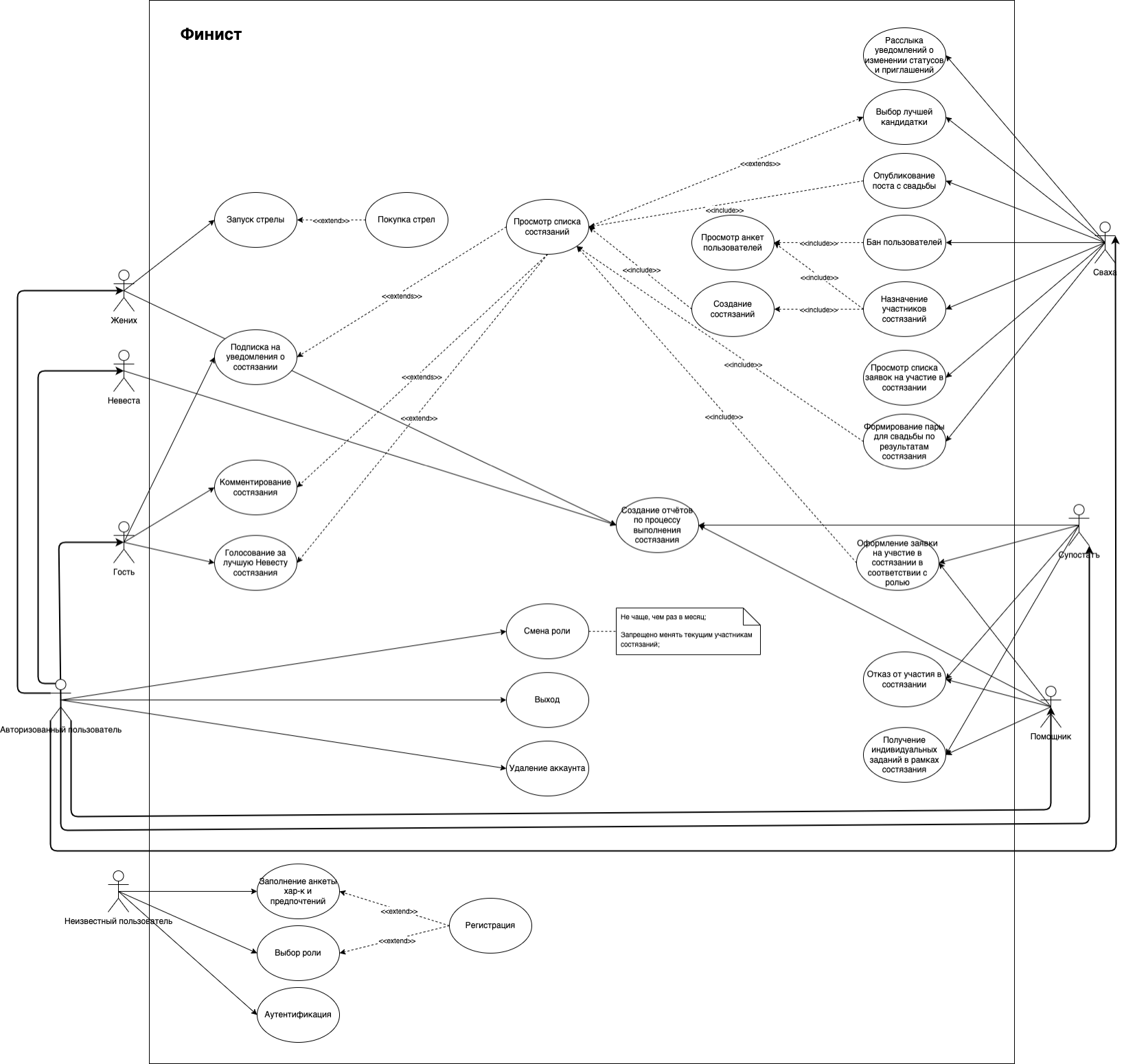
* FreeBSD 13
* 128GB ram / 560GB space / Intel Xeon E5-2643 3.3GHz / 16 cores
* PSQL 14
* Java 17
* Фронтенд будет упакован в jar (war), сервер бекенда будет использоваться как сервер для фронтенда
* Firefox 117, Chrome 117, Safari 17
* Стабильное интернет-соединение (LTE минимум)

Ни один из useCase-ов не создаёт особых архитектурных ограничений.

Система поддерживает как горизонтальное, так и вертикальное масштабирование средствами фреймворка Sprint, встроенного сервера и PSQL, путём развёртывания дополнительных инстансов программы и подключения к общей базе данных.

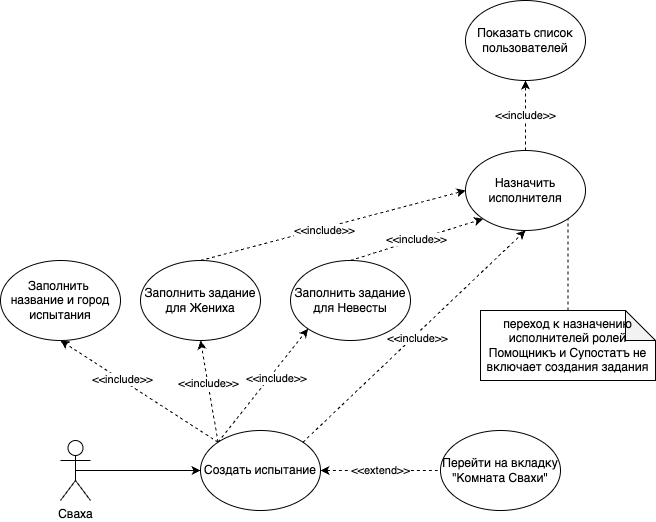
**4.** **Use-Case View**

Глобальная диаграмма UseCase:

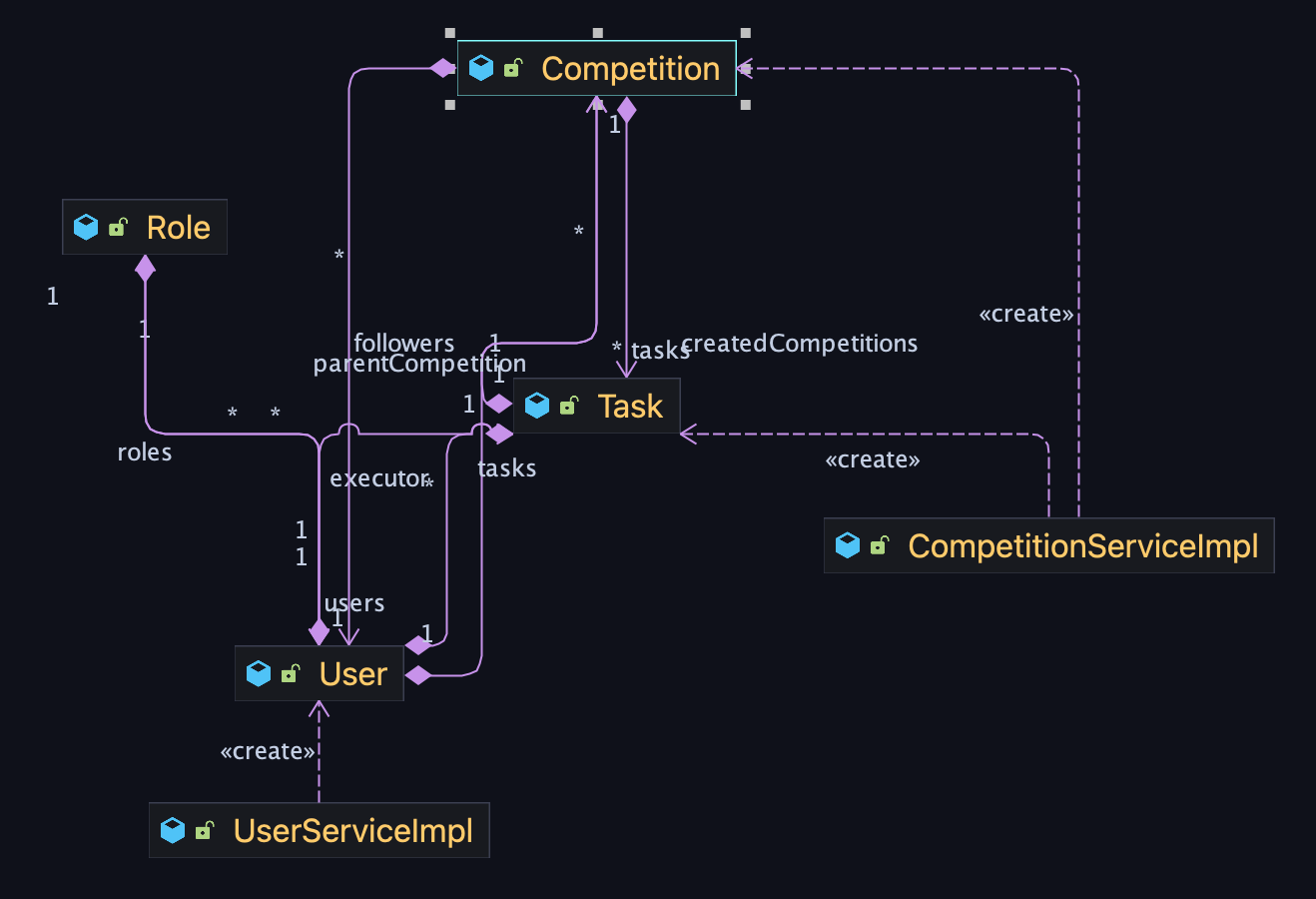


**18. Создание испытания**

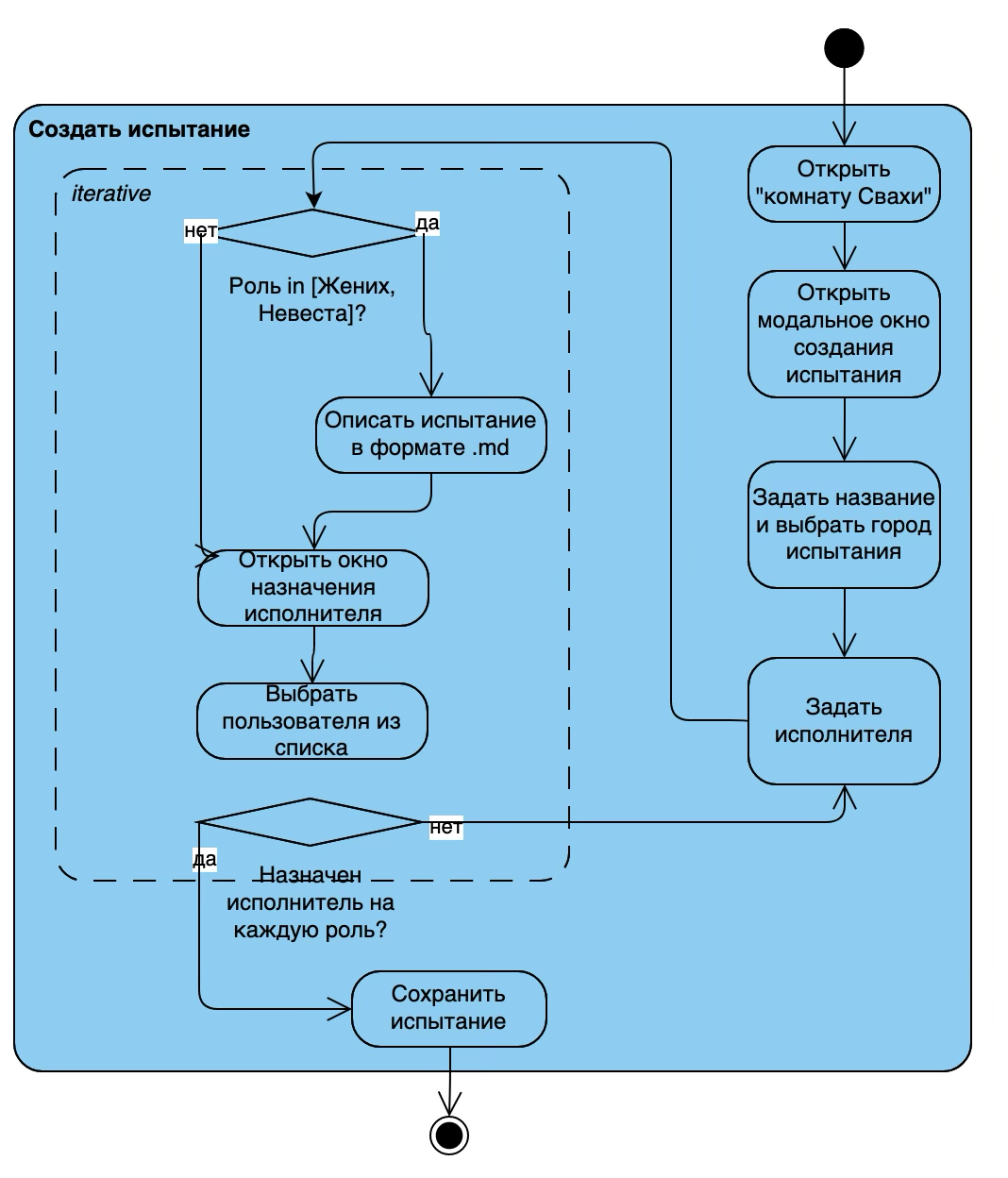
*UseCase:*

****

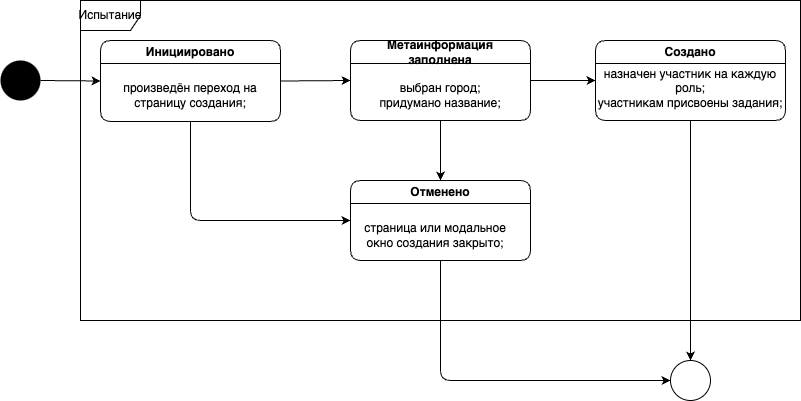
*Class:*

****

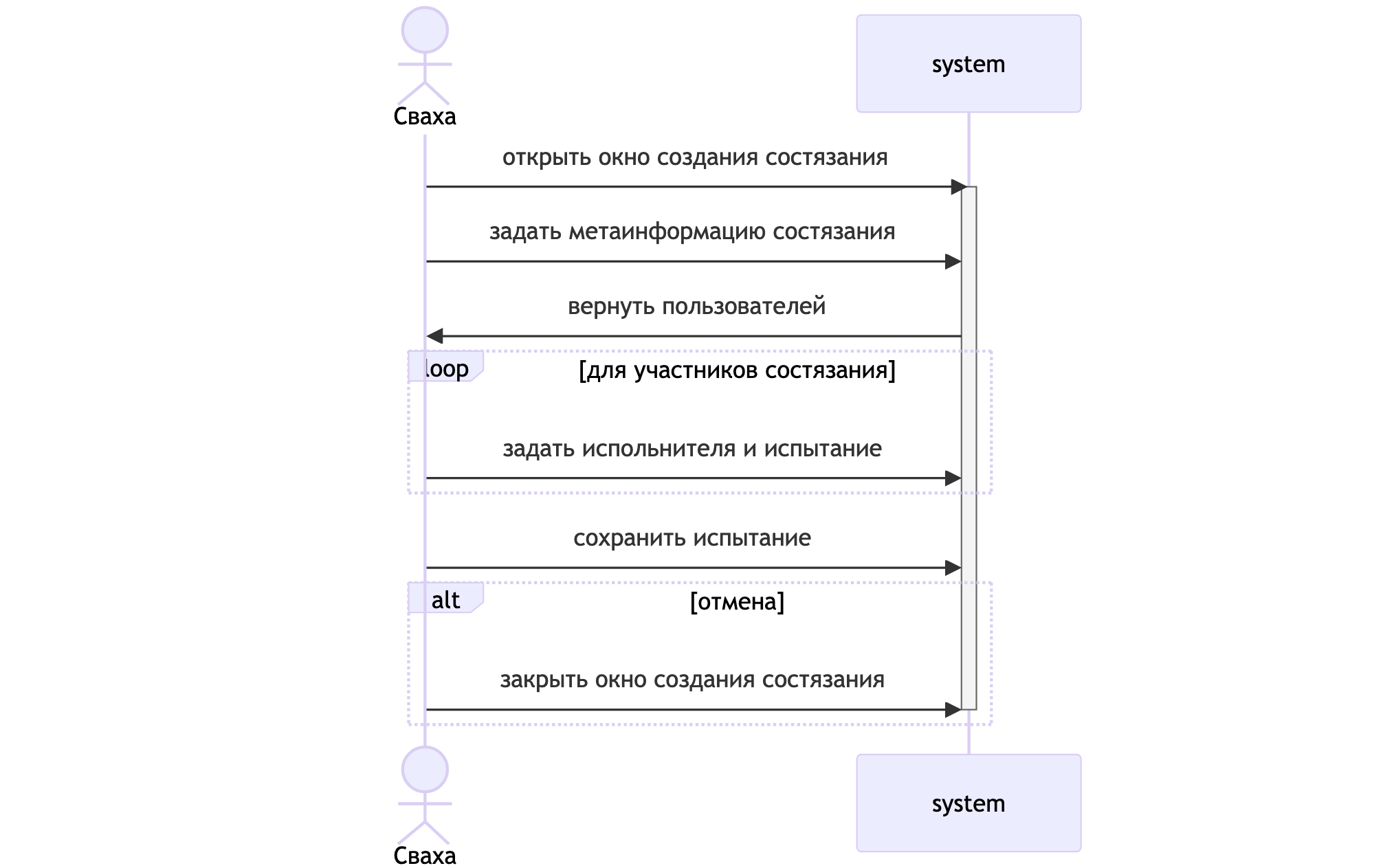
*Activity:*

**

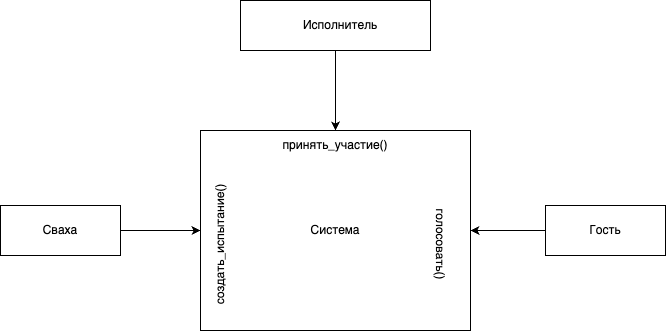
*State Machine:*

**

*Sequence:*

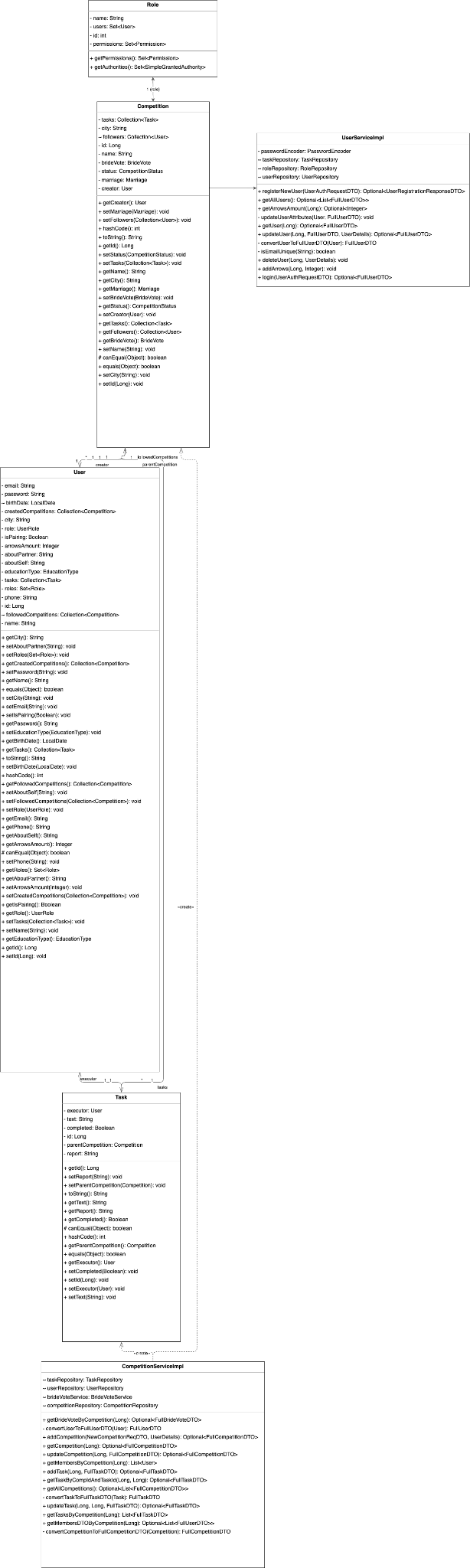
**

*Cooperative:*

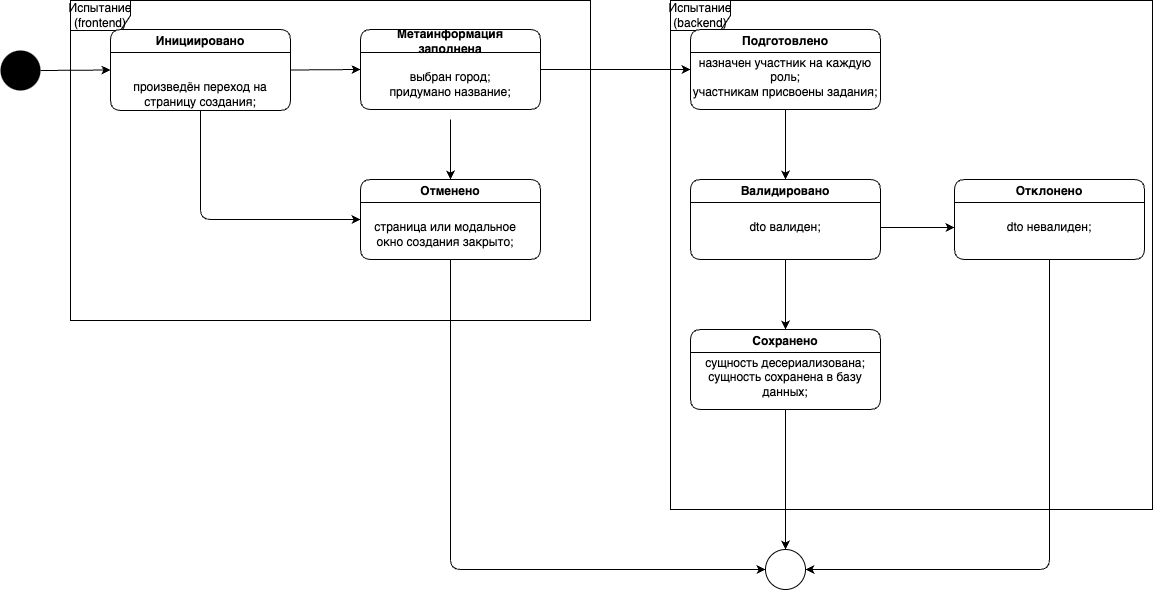
**

**5.** **Logical View**

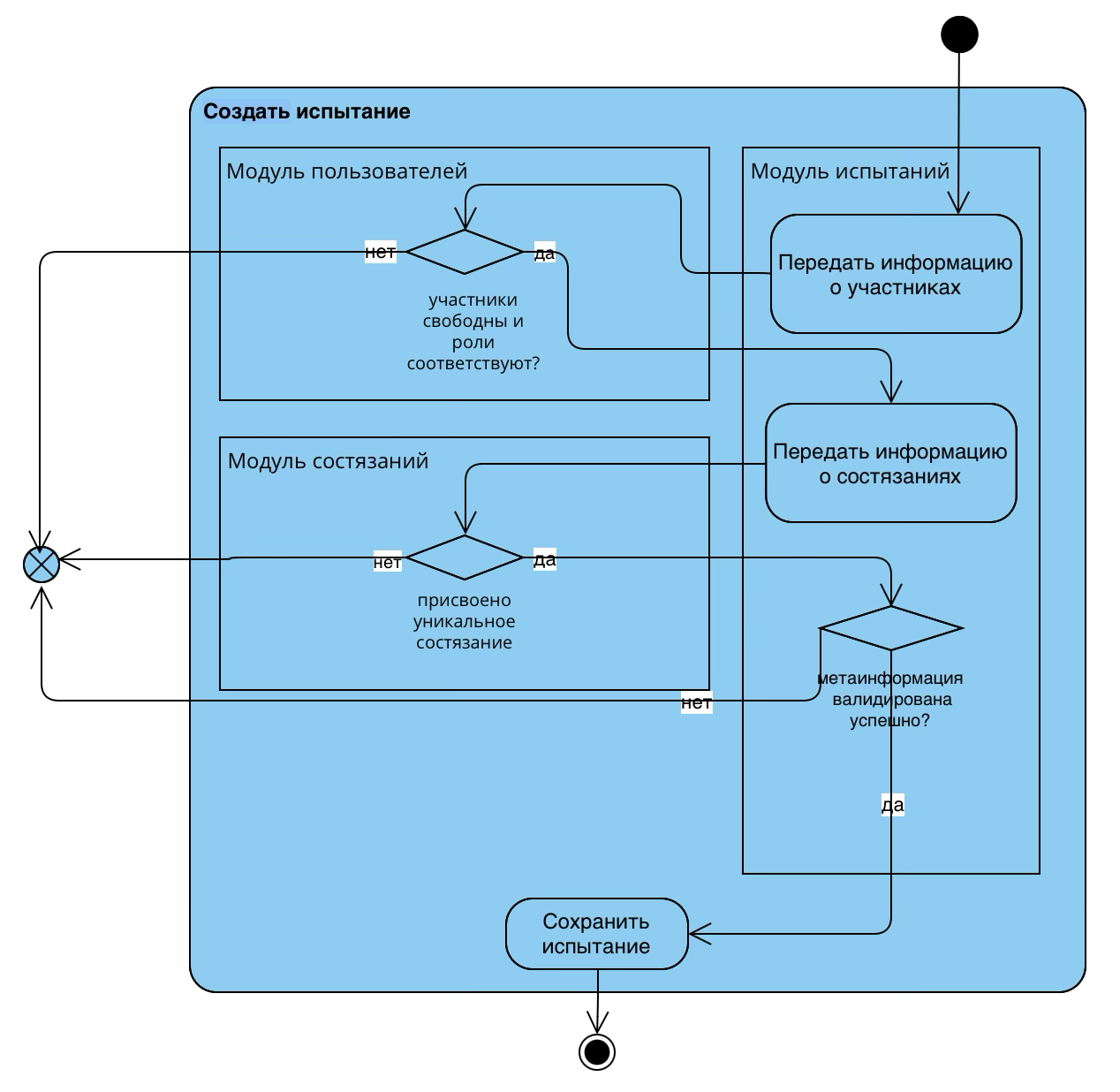
*Class:*

**

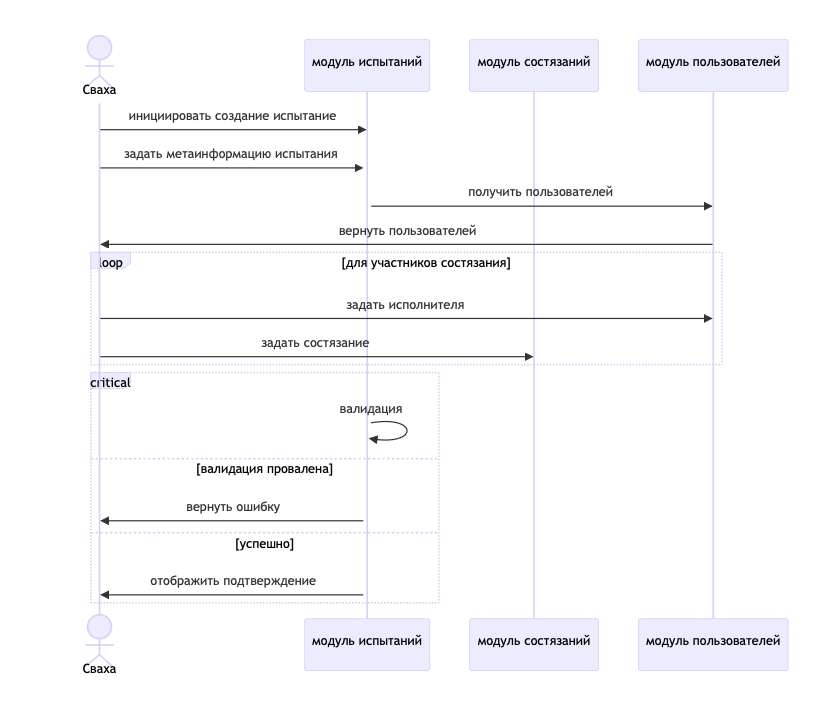
*State Machine:*

**

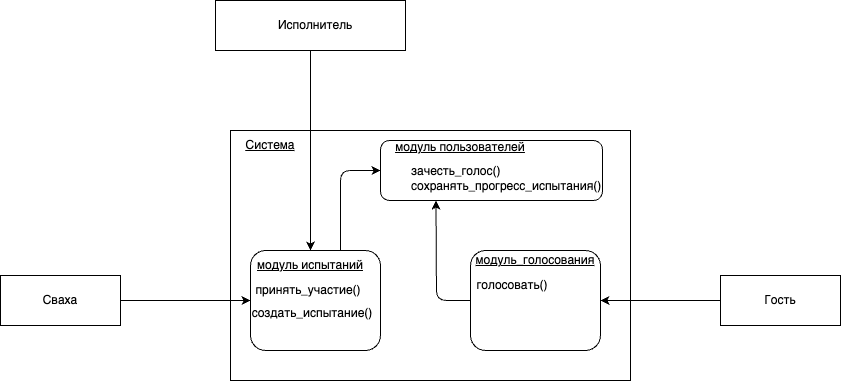
*Activity:*

**

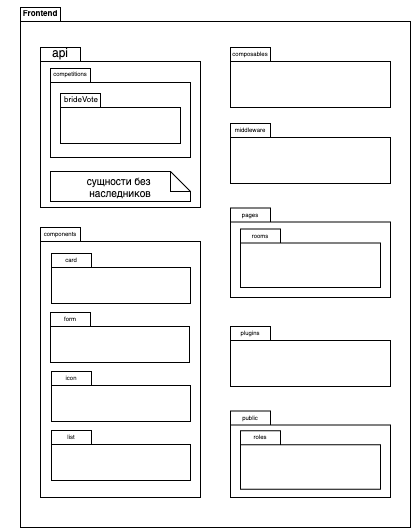
*Sequence:*

**

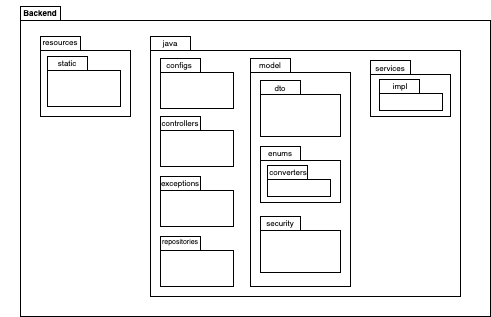
*Cooperative:*

**

*Package (frontend):*

**

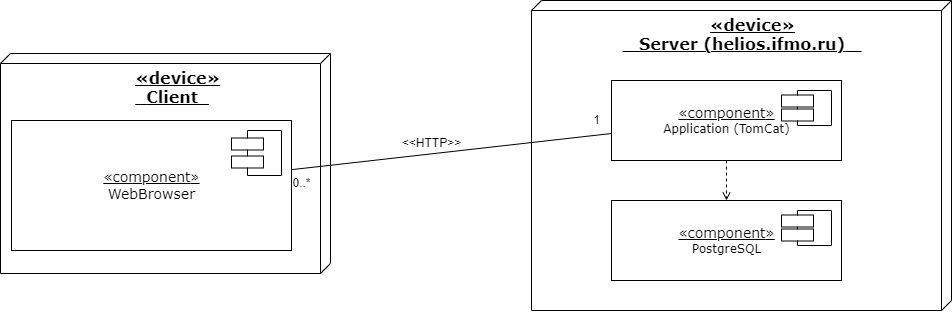
*Package (backend):*

**

**6.** **Process View**

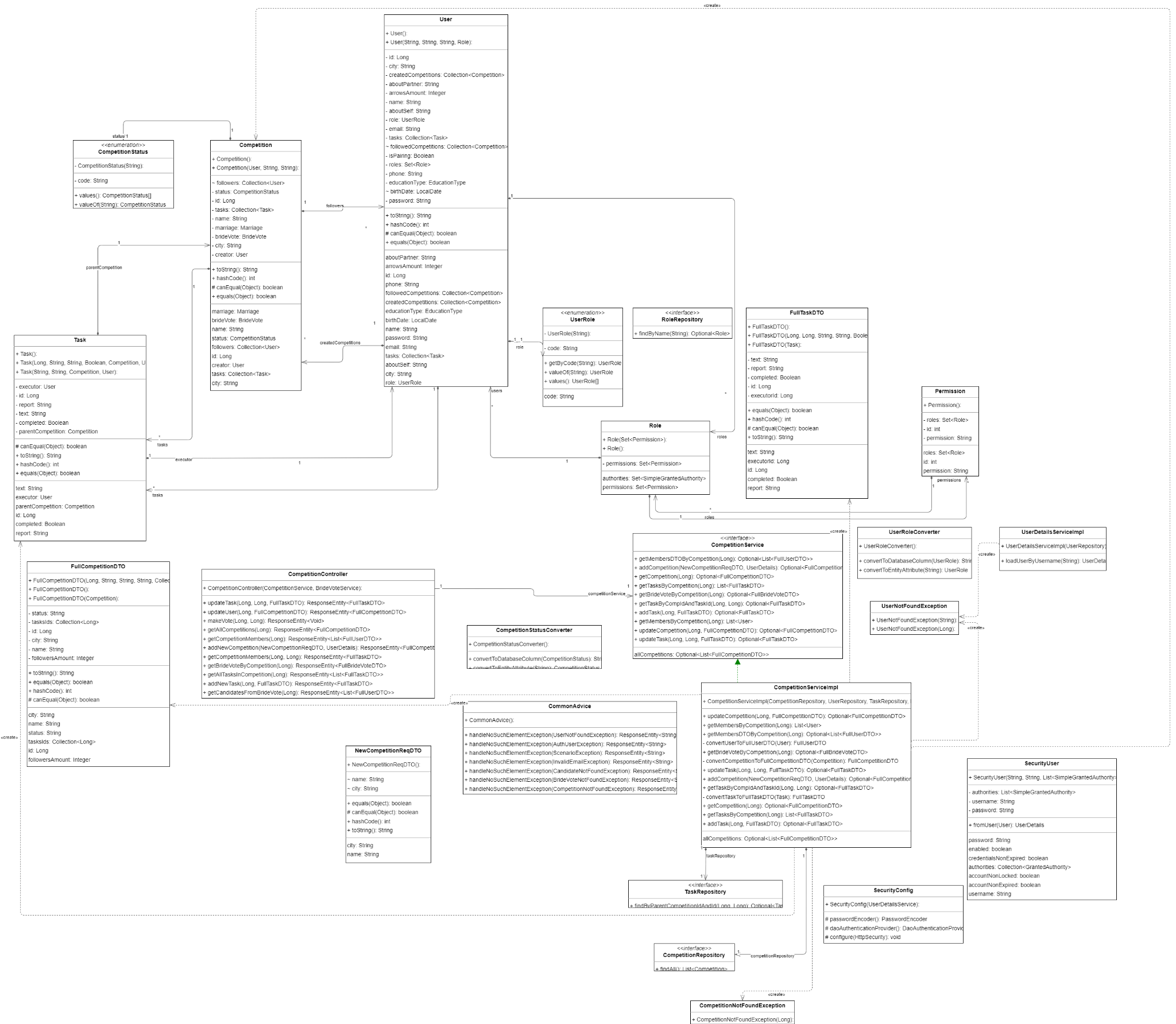
В наших usecase нет привязки к определенным моментам времени. Usecase-ы одним пользователей являются “callback-ом” usecase других, но задачи, являющиеся темпорально-зависимыми отсутствуют.

**7.** **Deployment View**

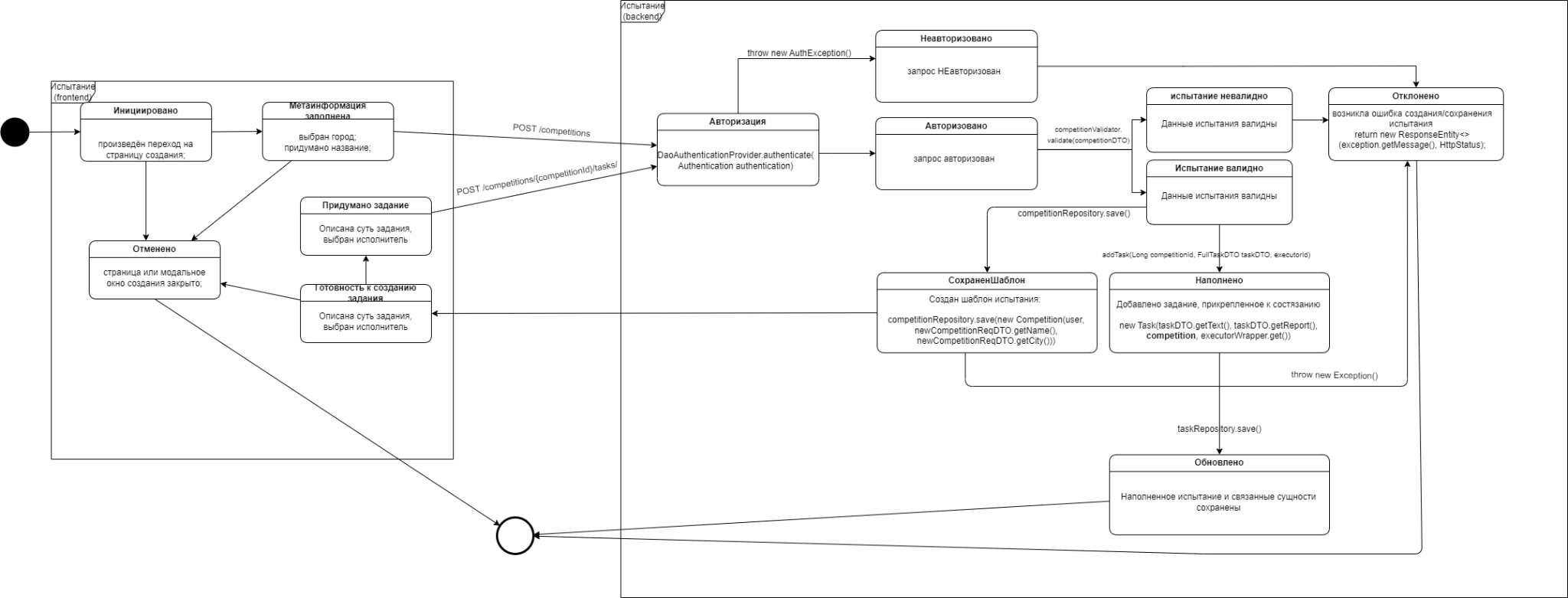
****

**8.** **Implementation View**

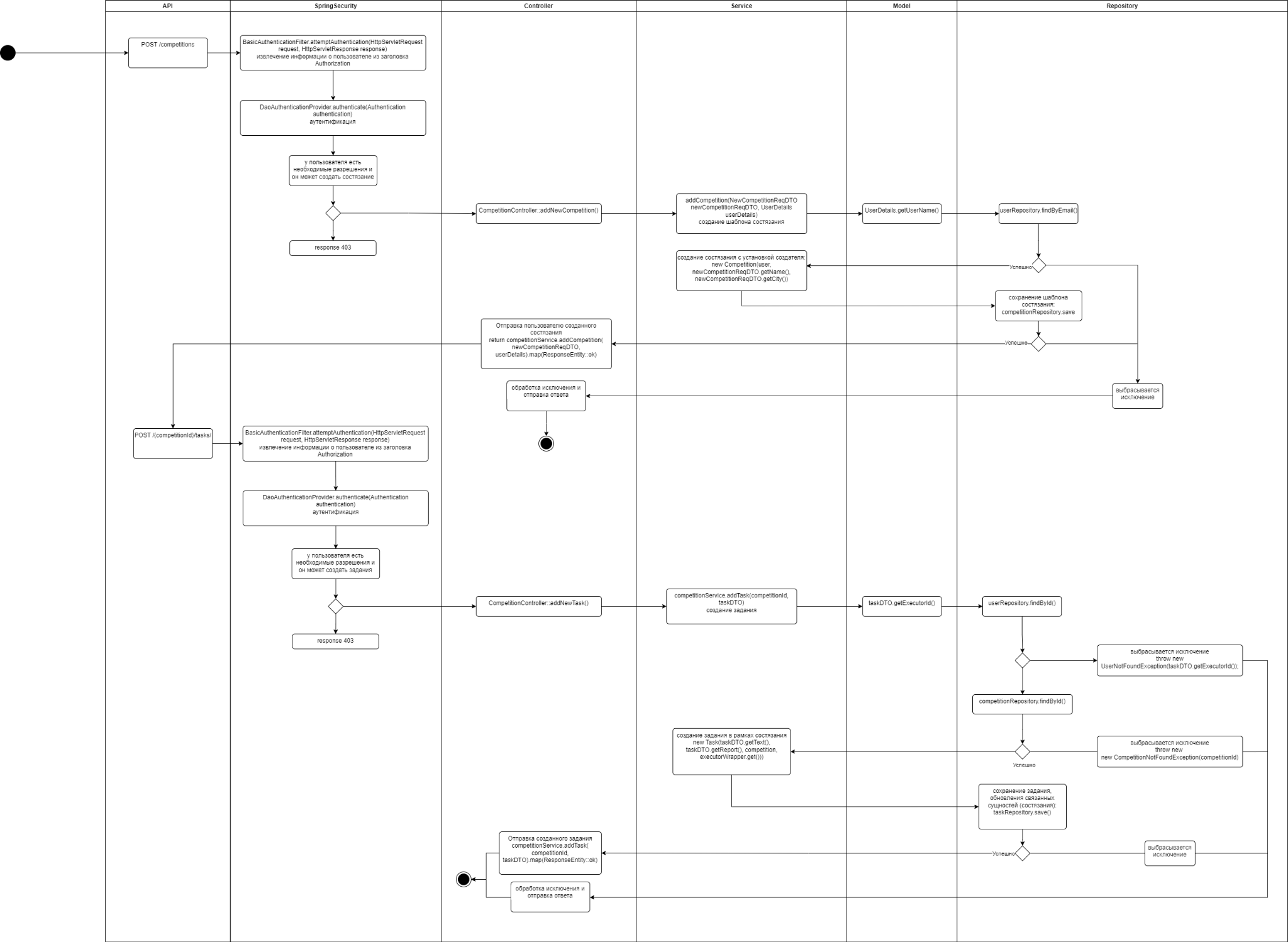
***Class Diagram***

******

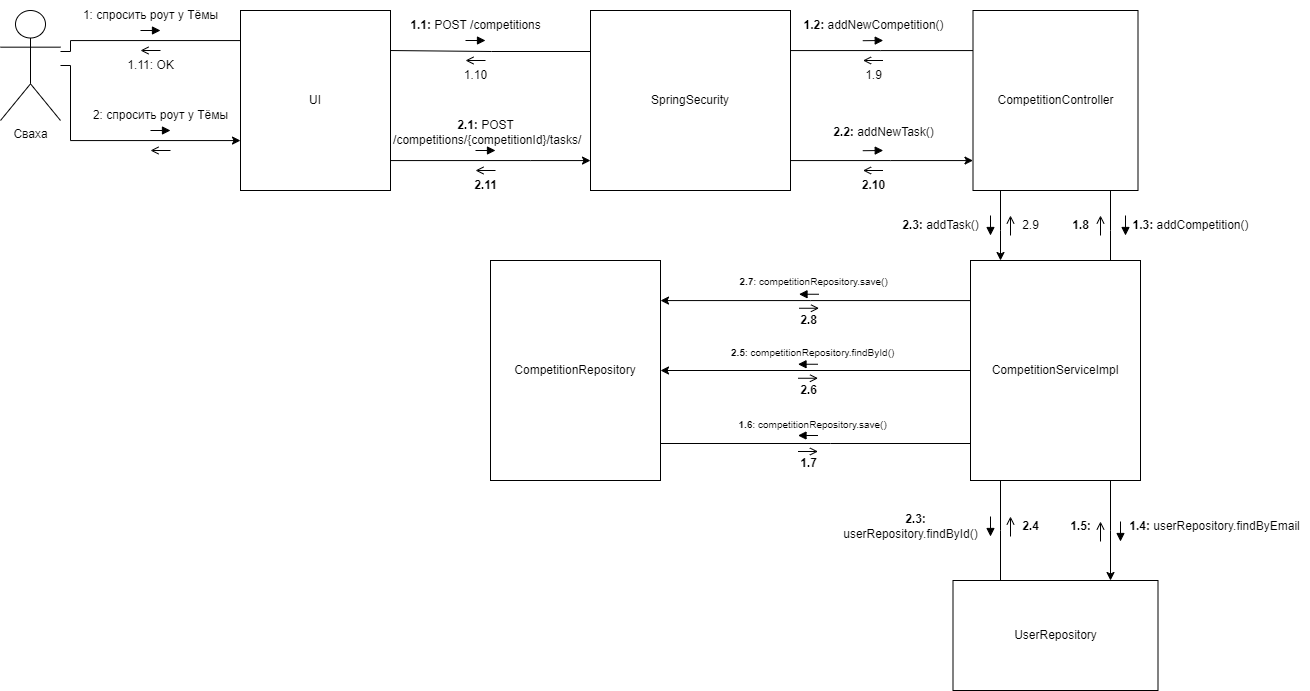
**State machine**

****

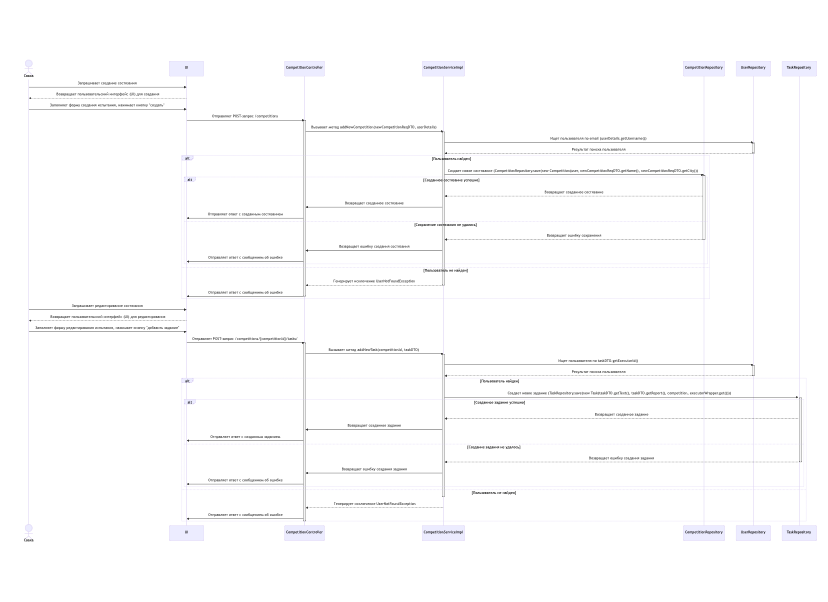
**Activity Diagram**

****

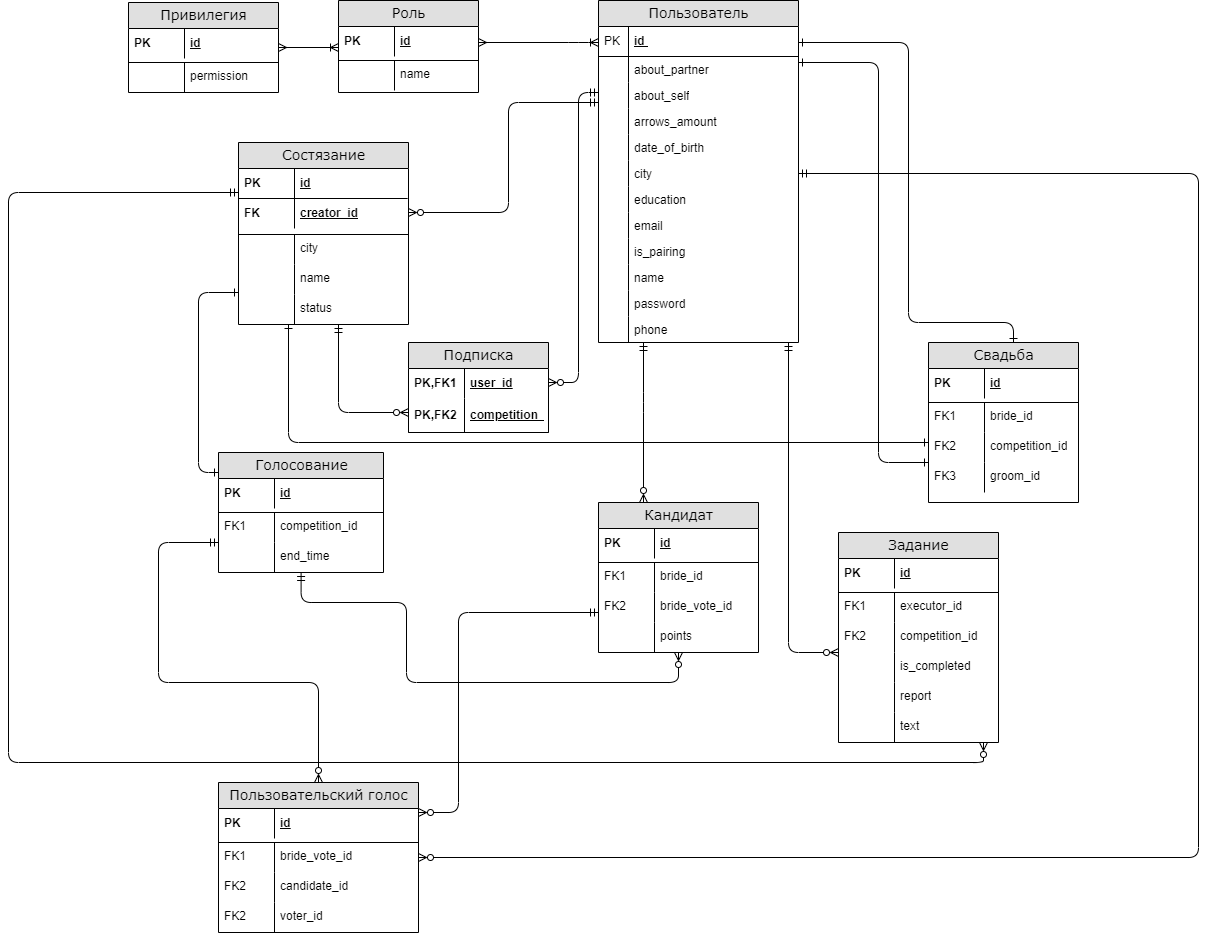
**Cooperative**

****

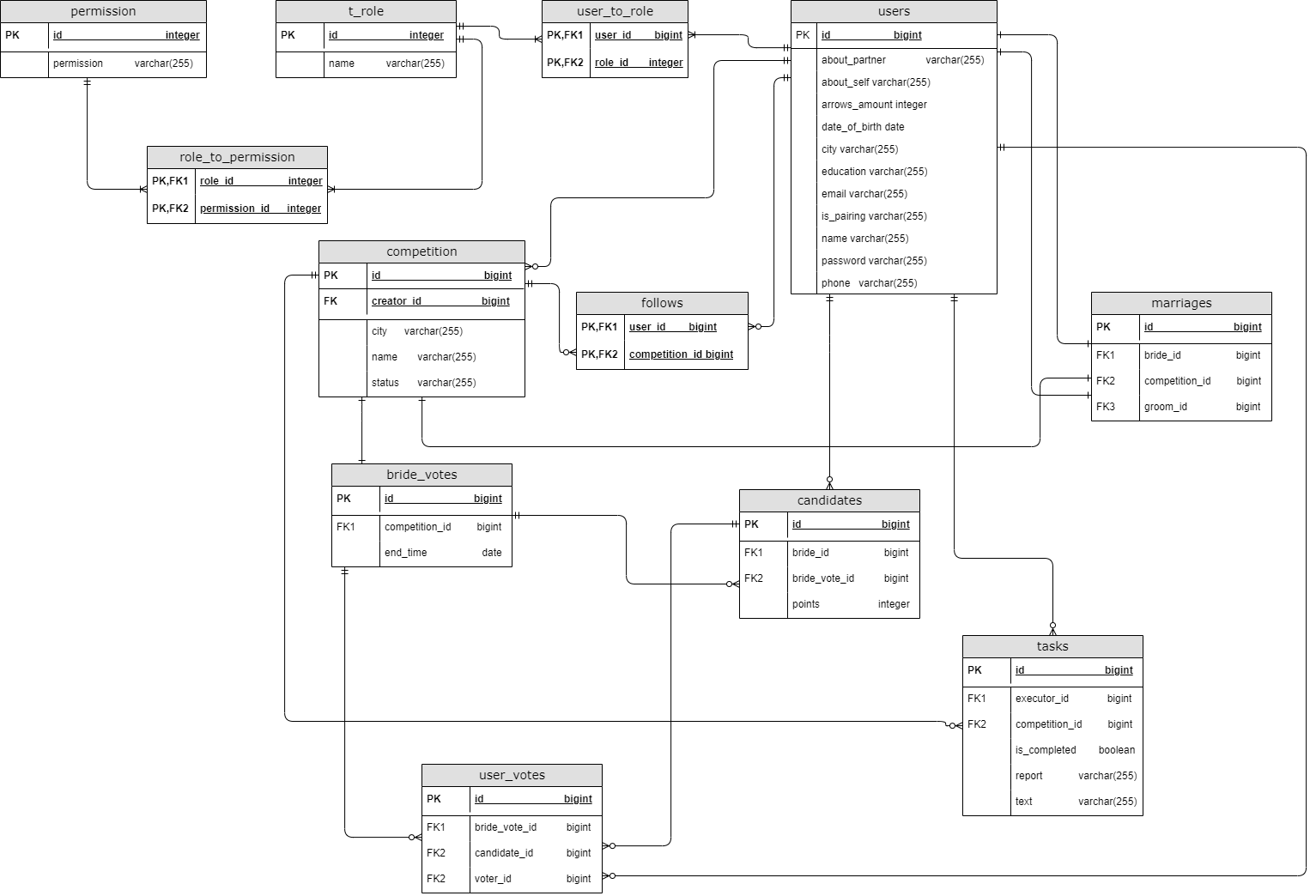
**Sequence**

****

**Инфологическая модель базы данных (ER)**

****

**Даталогическая модель базы данных**



**9.** **Size and Performance (Производительность)**

* Минимальное количество работающих пользователей: min[одновременно работающие пользователи] = **3** (роли Добрый молодец, Красна девица, Сваха, Супостатъ).
* Максимальное количество работающих пользователейmax[одновременно работающие пользователи] = **200000** (Ближайший конкурент - Tinder - имеет базу в [7000000 пользователей на территории РФ](https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%A1%D0%BE%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%81%D0%B5%D1%82%D0%B8_(%D1%80%D1%8B%D0%BD%D0%BE%D0%BA_%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D0%B8)). [Количество браков в год приблизительно такое же - 800000](https://www.fontanka.ru/2022/12/30/71941148/) (т.е. минимум 1600000 человек без учёта гостей и родственников). Т.е. будем считать, что гипотетически, каждый 2-ой брак может быть заключен с помощью нашего сервиса, тем не менее, его использование требует гораздо более активных действий, чем свайп в приложении для знакомств, будем считать, что 200000 - это гипотетический предел с огромным запасом с учётом зрителей, которых будет больше участников.
* Время отклика не должно превышать 2 секунды

**10.** **Quality (Качество)**

* Мобильная и десктопная версии верски веб-клиента
* Среднее время выполнения типовых задач = 2 минуты
* Время обучения для превращения в опытного пользователя (роль Сваха) = 1 день
* Достижения желаемого действия должно быть за минимальное количество кликов (<=10)
* Так как система не манипулирует жизненно важными данными и не управляет критическими процессами, требования к надёжности можно описать как “стандартные”: применение шифрования паролей (**SHA-1, MD-5, AES-128**), использование стабильных (**release**) версий программных библиотек, аутентификация пользователей по логину и паролю.
* Сервис не представляет доступа к каким-то жизненно важным ресурсам, то и его простой не является критичным, поэтому, будет считать, что коэффициент доступности **96%** даже избыточен. Таким образов, сервис будет недоступен всего около 350 часов в год (менее часа в день).