Министерство образования Республики Беларусь

Учреждение образование

«Белорусский государственный университет

информатики и радиоэлектроники»

**ОТЧЕТ**

**по производственной практике**

Место прохождения практики:

Общество с ограниченной ответственностью

«Фабрика инноваций и решений»

Сроки прохождения практики: с 10.06.2024 по 07.07.2024

Руководитель практики от Студент группы 151003

предприятия: \_\_\_\_\_\_\_\_\_\_\_\_Барановский Р.А.

\_\_\_\_\_\_\_\_\_ Кисляк Д.С. (подпись студента)

(подпись руководителя) Руководитель практики от БГУИР

Можей Н.П.

Минск 2024

Содержание

[Введение 4](#_Toc170835765)

[1 Общие сведения о предприятии 5](#_Toc170835766)

[1.1 О «Фабрика инноваций и решений» 5](#_Toc170835767)

[1.2 Пакет прикладных программ, используемых на предприятии 5](#_Toc170835768)

[1.2.1 Confluence 5](#_Toc170835769)

[1.2.2 Google Docs 6](#_Toc170835770)

[1.2.3 Google Meet 6](#_Toc170835771)

[1.2.4 Google Chat 6](#_Toc170835772)

[1.2.5 Skype…………………………………………………………………………7](#_Toc170835773)

[1.2.6 Zoom…………………………………………………………………………7](#_Toc170835774)

[1.2.7 Microsoft Teams 7](#_Toc170835775)

[1.2.8 Jira……………………………………………………………………………7](#_Toc170835776)

[1.2.9 Diagrams.net 8](#_Toc170835777)

[2 Методы разработки программного обеспечения 9](#_Toc170835778)

[2.1 Процесс разработки программного продукта 9](#_Toc170835779)

[2.1.1 Agile методология 9](#_Toc170835780)

[2.1.2 Scrum методология 9](#_Toc170835781)

[2.1.3 Kanban методология 10](#_Toc170835782)

[2.2 Технологии разработки 12](#_Toc170835783)

[2.2.1 Spring Framework 12](#_Toc170835784)

[2.2.2 Hibernate 13](#_Toc170835785)

[2.2.3 Junit…………………………………………………………………………14](#_Toc170835786)

[2.2.4 Apache Maven 14](#_Toc170835787)

[2.2.5 Thymeleaf 16](#_Toc170835788)

[3 выполнение индивидуального задания 17](#_Toc170835789)

[3.1 Индивидуальное задание 17](#_Toc170835790)

[3.2 Используемые технологии 17](#_Toc170835791)

[3.3 Архитектура проекта 17](#_Toc170835792)

[3.4 Функциональные требования 18](#_Toc170835793)

[3.5 Нефункциональные требования 18](#_Toc170835794)

[Заключение 19](#_Toc170835795)

[Приложение А 21](#_Toc170835796)

[4 Формирование технической задачи дипломного проекта 24](#_Toc170835797)

[4.1 Определение целей и задач проекта 24](#_Toc170835798)

[4.1.1 Цель проекта 24](#_Toc170835799)

[4.1.2 Задачи проекта 24](#_Toc170835800)

[4.2 Функциональные требования 24](#_Toc170835801)

[4.2.1 Основные модули 24](#_Toc170835802)

[4.2.2 Пользовательские роли 24](#_Toc170835803)

[4.3 Нефункциональные требования 24](#_Toc170835804)

[4.3.1 Производительность 24](#_Toc170835805)

[4.3.2 Надёжность 25](#_Toc170835806)

[4.3.3 Безопасность 25](#_Toc170835807)

[4.3.4 Интерфейс 25](#_Toc170835808)

Введение

Производственная практика является важным компонентом образовательного процесса в высших учебных заведениях, обеспечивающим студентам возможность применить теоретические знания на практике и получить ценный профессиональный опыт в реальных условиях.

Целью производственной практики является закрепление и применение на практике теоретических знаний, полученных в ходе обучения, а также приобретение необходимых профессиональных навыков и компетенций.

На время прохождения производственной практики были поставлены следующие задачи:

1. Ознакомиться и описать структуру предприятия, вид деятельности и пакет прикладных программ, используемых на предприятии;
2. Изучить и описать методы разработки ПО различного функционального назначения;
3. Выполнить индивидуальное задание, предложенное руководителем от предприятия;
4. Собрать материал для формирования технической задачи, которую планируется решать в дипломном проекте.

# Общие сведения о предприятии

## О «Фабрика инноваций и решений»

Innowise Group – одна из наиболее опытных белорусских компаний в области полного цикла разработки, внедрения и поддержки ПО, бизнес-анализа, дизайна, тестирования и управления проектами.[7]

Компания основана в 2007г. За это время реализовано 500+ проектов для клиентов из из США, Канады, Германии, Японии (всего более 30 стран).

Центры разработки Innowise находятся в 6 городах Беларуси (Минск, Витебск, Гомель, Брест, Могилев и Полоцк), а также в Польше, Литве, Украине и Грузии.

Компания работает более чем с 20 бизнес-доменами (medicine, fintech, entertainment, eCommerce и пр.), создает решения с использованием AI, Machine Learning, AR/VR.

Также компания создает собственные продукты. Проект VOKA.IO, основанный на применении технологий Mixed Reality в области хирургии, победил в категории “Здоровье” всемирного конкурса инновационных проектов World Summit Awards 2019 и получил международное признание от врачей всего мира.

## Пакет прикладных программ, используемых на предприятии

Для обеспечения качественных услуг команда Innowise использует самые передовые инструменты и технологии для повышения эффективности и качества работы. Также компания постоянно следит за развитием отрасли и регулярно внедряет новые решения. Рассмотрим основные прикладные программы, применяемые на предприятии.

### Confluence

Confluence – популярный корпоративный инструмент, который широко применяется для создания технических текстов, проектной документации и информационных порталов. Подходит рабочим командам независимо от их размера и типа. Это программное обеспечение, с помощью которого можно создавать и редактировать страницы с текстом, изображениями, инфографикой и прикрепленными файлами.

Информация в Confluence организуется в логические пространства (spaces), которые можно структурировать по проектам, отделам или другим категориям. Внутри каждого пространства создаются страницы и подстраницы, что помогает систематизировать данные понятным для пользователей образом.

Одно из ключевых преимуществ Confluence - его интеграция с широким спектром других приложений, таких как Jira, Trello и другие продукты Atlassian. Это позволяет легко связывать задачи, проекты и документацию, обеспечивая целостный рабочий процесс. Мощные функции поиска также помогают быстро находить нужные страницы и документы.

Использование Confluence способствует лучшему обмену знаниями и повышению общей эффективности работы сотрудников. Это один из ключевых инструментов, обеспечивающих согласованность и прозрачность деятельности организаций. В целом, Confluence зарекомендовал себя как эффективный инструмент для совместной работы над контентом, структурирования информации и интеграции данных в рамках организации.

### Google Docs

Google Docs – облачный текстовый редактор, разработанный компанией Google. Он позволяет пользователям создавать, редактировать и хранить документы в интернете. Главное преимущество Google Docs - поддержка совместной работы. Несколько человек могут одновременно вносить правки в один и тот же документ, и при этом все участники будут видеть изменения друг друга в режиме реального времени.

Такая функциональность делает Google Docs удобным инструментом для коллективной работы над текстовыми материалами. Пользователи могут легко комментировать документы, предлагать правки и обсуждать изменения прямо внутри приложения. Это ускоряет процесс создания и согласования контента в командах.

### Google Meet

Google Meet – сервис видео-телефонной связи и видеоконференций, разработанный компанией Google. Он позволяет пользователям проводить онлайн-встречи, совещания и интерактивные сеансы в режиме реального времени. Также поддерживает демонстрацию рабочего стола одного пользователя другим.

### Google Chat

Google Chat – коммуникационный сервис, разработанный компанией Google. Изначально он был разработан для рабочих групп и бизнес-среды, но затем стал доступен и для обычных пользователей. Он предоставляет личные сообщения между пользователями, групповые беседы и чат-группы, которые позволяют пользователям не только общаться в чате, но и создавать и назначать задания, а также обмениваться файлами в одном месте. Доступ к сервису можно получить через собственные веб-сайт и приложение, или через веб-сайт и приложение Gmail.

### Skype

Skype – популярное программное обеспечение для голосовой связи, видеозвонков и обмена сообщениями, разработанное компанией Microsoft. Это один из наиболее известных и широко используемых коммуникационных сервисов. Программа позволяет совершать конференц-звонки, видеозвонки, а также обеспечивает передачу текстовых сообщений (чат) и передачу файлов. Есть возможность вместе с изображением с веб-камеры передавать изображение с экрана монитора, а также создавать и отправлять видеосообщения пользователям настольных версий программы.

### Zoom

Zoom – высокотехнологичная платформа для видеоконференций и совместной работы, разработанная компанией Zoom Video Communications. Zoom быстро завоевала популярность и стала одним из ведущих сервисов для проведения онлайн-встреч, вебинаров и конференций.

Преимуществами Zoom можно назвать возможность простой аналитики и ведения статистики, возможность работать с презентациями во время звонков, а также наличие режима «белой доски».

### Microsoft Teams

Microsoft Teams – комплексное облачное решение для совместной работы и коммуникаций, разработанное компанией Microsoft. Являясь частью пакета Microsoft 365, Teams предлагает широкий спектр возможностей для повышения эффективности командной работы в организациях.

Teams позволяет пользователям обмениваться текстовыми сообщениями, проводить голосовые и видеозвонки, а также участвовать в групповых чатах. Также предоставляет встроенные инструменты для управления задачами, назначения ответственных, отслеживания прогресса и планирования. Поддерживает интеграцию с приложениями Microsoft Office и позволяет совместно редактировать файлы Word, Excel, PowerPoint и OneNote в режиме реального времени.

### Jira

Jira – система управления проектами и отслеживания задач, разработанная компанией Atlassian. Поддерживает управление проектами и задачами, позволяя создавать задачи, назначать ответственных за выполнение, устанавливать приоритеты и сроки, а также отслеживать ход выполнения работ. Позволяет гибко настраивать рабочие процессы, а также обладает функционалом, необходимым для применения Agile-подходов, таких как Scrum и Kanban.

Jira поддерживает интеграцию с различными инструментами разработки, управления кодом, CI/CD, а также с другими корпоративными системами. Благодаря своей всесторонней функциональности, гибкости и расширяемости, Jira стала одним из наиболее популярных решений для управления проектами и задачами, особенно среди организаций, использующих Agile-методологии.

### Diagrams.net

Diagrams.net – кроссплатформенное программное обеспечение для рисования графиков , разработанное на HTML5 и JavaScript. Его интерфейс можно использовать для создания диаграмм, таких как блок-схемы , каркасы , диаграммы UML, организационные диаграммы и сетевые диаграммы.

Интегрируется с [облачными сервисами](https://en.wikipedia.org/wiki/Cloud_services) для хранения данных, включая [Dropbox](https://en.wikipedia.org/wiki/Dropbox), [OneDrive](https://en.wikipedia.org/wiki/OneDrive), [Google Drive](https://en.wikipedia.org/wiki/Google_Drive), [GitHub](https://en.wikipedia.org/wiki/GitHub) и [GitLab.com](https://en.wikipedia.org/wiki/GitLab). Доступен в виде [плагина](https://en.wikipedia.org/wiki/Plug-in_(computing)) для встраивания веб-приложения в такие платформы, как [NextCloud](https://en.wikipedia.org/wiki/Nextcloud" \o "Nextcloud), [MediaWiki](https://en.wikipedia.org/wiki/MediaWiki" \o "МедиаВики), [Notion](https://en.wikipedia.org/wiki/Notion_(productivity_software)" \o "Notion (программное обеспечение для повышения производительности)), [Confluence](https://en.wikipedia.org/wiki/Confluence_(software)" \o "Слияние (программное обеспечение)) и [Jira](https://en.wikipedia.org/wiki/Jira_(software)" \o "Jira (программное обеспечение)).

Draw.io позволяет экспортировать готовые диаграммы в различные форматы файлов, включая PNG, JPEG, PDF, SVG и другие, что делает их удобными для публикации, вставки в презентации или документацию.

Использование этих программных продуктов позволяет команде Innowise эффективно управлять рабочим процессов и способствует высокому качеству разрабатываемого программного обеспечения .

# Методы разработки программного обеспечения

## Процесс разработки программного продукта

В компании Innowise придерживаются Agile методологии разработки, что позволяет полностью информировать клиентов о ходе выполнения проекта, повышает уровень вовлеченности и удовлетворенности клиентов, способствует созданию доверительных отношений.

К отдельным Agile-подходам относятся Scrum и Kanban. Методология Scrum используется в качестве основы для разработки проектов, в то время как Kanban служит вспомогательной структурой.

### Agile методология

Agile - это гибкий подход к управлению проектами, основанный на непрерывном улучшении, адаптивности, скорости и тесном взаимодействии с клиентами.[8] Он возник в ответ на ограничения традиционных методов, таких как Waterfall, которые часто не могут быстро реагировать на изменения и требуют значительных временных затрат на планирование и документацию.

Ключевые принципы Agile включают в себя акцент на людях и коллаборации вместо процессов и инструментов, приоритет работающего продукта над полной документацией, сотрудничество с заказчиком вместо следования контракту, а также готовность к изменениям вместо строгого следования первоначальному плану. Эти принципы способствуют более гибкому и адаптивному подходу к управлению проектами, что особенно важно в быстро меняющихся условиях рынка и технологий.

Преимущества использования Agile включают улучшение качества продукта за счет постоянного тестирования и итераций, быстрое выявление и исправление ошибок, повышение удовлетворенности клиентов благодаря их вовлечению в процесс разработки, снижение рисков за счет возможности адаптации проекта к изменяющимся требованиям, а также более эффективную, мотивированную и вовлеченную работу команд.

В Innowise применяются различные инструменты для следования Agile методологии, такие как Jira, Trello, Tableau, Power BI, GitLab, GitHub, BitBucket, Jenkins и GitLab CI/CD, которые позволяют управлять проектами, отслеживать прогресс, осуществлять контроль версий и непрерывную интеграцию/доставку.

### Scrum методология[9]

Scrum - это одна из популярных методологий Agile, используемая для управления сложными проектами. Она основана на итеративном и инкрементальном подходе, что помогает командам быстро реагировать на изменения и создавать ценный продукт.

Ключевые роли в Scrum:

* владелец продукта - управляет бэклогом;
* скрам-мастер - обеспечивает соблюдение принципов Scrum и помогает команде;
* команда разработки - непосредственно работает над продуктом.

Процесс Scrum включает:

* спринты - фиксированные периоды времени, в которые команда работает над конкретными целями;
* планирование спринта, ежедневные скрам-встречи, спринт-ревью и ретроспективы.

Преимущества Scrum:

* прозрачность и видимость прогресса;
* адаптивность к изменениям;
* повышение мотивации и вовлеченности команды;
* улучшение качества продукта;
* снижение рисков и повышение удовлетворенности клиентов.

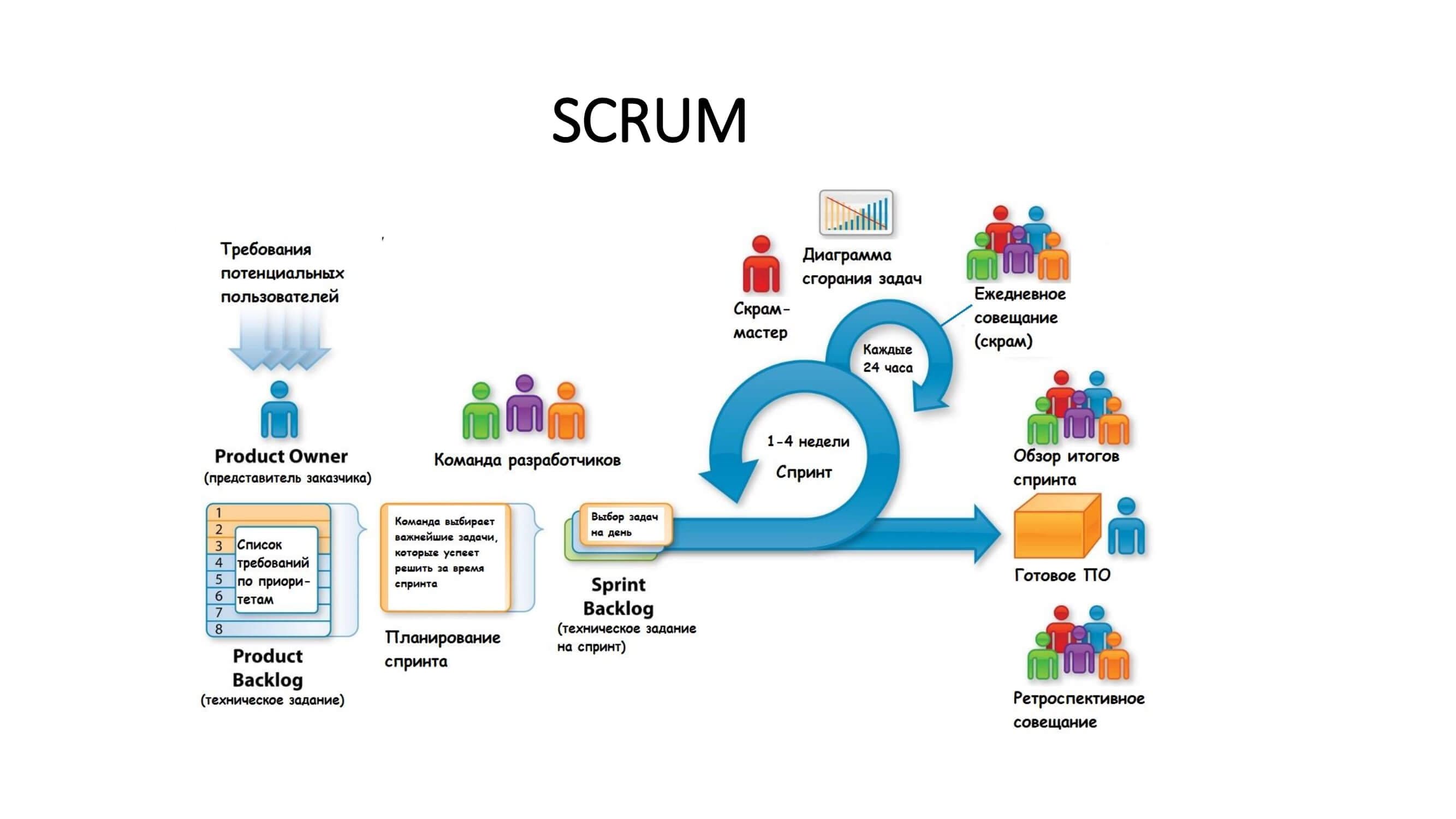


Рисунок 1 – Жизненный цикл разработки программного продукта по Scrum методике

### Kanban методология[9]

Kanban - это методика управления проектами, которая помогает командам визуализировать и оптимизировать рабочие процессы. Изначально она была разработана компанией Toyota для повышения эффективности производства, но сейчас широко применяется в различных областях, включая разработку ПО, оказание ИТ-услуг и маркетинг.

В основе Kanban лежит канбан-доска со столбцами, соответствующими этапам выполнения задач, например "Запланировано", "В работе" и "Выполнено". Задачи отображаются в виде карточек, которые перемещаются по доске по мере их прохождения через этапы.

Ключевой принцип Kanban - ограничение незавершенной работы (WIP). Это означает, что в каждом столбце доски может находиться только определенное количество карточек, чтобы команда могла сосредоточиться на завершении текущих задач перед началом новых. Это помогает избежать перегрузки и способствует более равномерному распределению работы.

Kanban также поощряет постоянное улучшение процессов - команды регулярно анализируют и оптимизируют свои рабочие потоки, выявляя и устраняя узкие места. Визуализация рабочего процесса помогает в этом, делая его прозрачным для всех заинтересованных сторон.

Основные преимущества Kanban - повышение прозрачности, ответственности, производительности и качества работы, а также гибкость в адаптации к изменениям. В итоге это приводит к более высоким результатам и удовлетворенности клиентов.

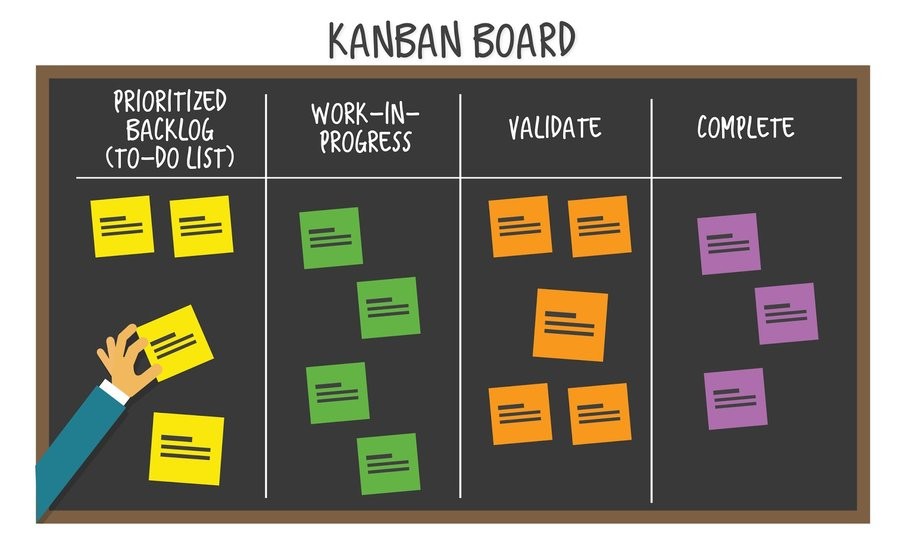


Рисунок 2 – Стадии выполнения задач по методике Kanban

Опыт показывает, что использование Agile подходов существенно улучшает эффективность работы команды и качество выпускаемых продуктов, что является ключевым фактором для успеха проектов в динамичной среде разработки программного обеспечения.

## Технологии разработки

В компании Innowise при разработке программного обеспечения широко применяются различные фреймворки и технологии из экосистемы Java. Этот мощный набор инструментов позволяет создавать высококачественные и масштабируемые приложения, удовлетворяющие требования клиентов. Помимо значительного опыта в области традиционных Java-технологий, компонентов и фреймворков, инженеры компании постоянно отслеживают выход новых версий JDK, технологий, компонентов и фреймворков, внедряя передовые решения.

### Spring Framework[10]

Spring Framework – универсальный [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA) [с открытым исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для [Java](https://ru.wikipedia.org/wiki/Java)-платформы. Spring предоставляет большую свободу Java-разработчикам в проектировании. Кроме того, он предоставляет хорошо документированные и лёгкие в использовании средства решения проблем, возникающих при создании приложений корпоративного масштаба.

Между тем, особенности ядра Spring применимы в любом Java-приложении, и существует множество расширений и усовершенствований для построения [веб-приложений](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) на [Java Enterprise платформе](https://ru.wikipedia.org/wiki/J2EE). По этим причинам Spring приобрёл большую популярность и признаётся разработчиками как стратегически важный фреймворк.

Spring, вероятно, наиболее известен как источник расширений (features), нужных для эффективной разработки сложных бизнес-приложений вне тяжеловесных программных моделей, которые исторически были доминирующими в промышленности. Ещё одно его достоинство в том, что он ввел ранее неиспользуемые функциональные возможности в сегодняшние господствующие методы разработки, даже вне платформы Java.

Этот фреймворк предлагает последовательную модель и делает её применимой к большинству типов приложений, которые уже созданы на основе платформы Java. Считается, что Spring реализует модель разработки, основанную на лучших стандартах индустрии, и делает её доступной во многих областях Java.

Spring может быть рассмотрен как совокупность множества меньших фреймворков. Большинство этих фреймворков может работать независимо друг от друга, однако они обеспечивают большую функциональность при совместном их использовании. Эти фреймворки делятся на структурные элементы типовых комплексных приложений. Ниже приведены некоторые из них:

* [Inversion of Control](https://ru.wikipedia.org/wiki/Inversion_of_Control)-контейнер: делегированное (фреймворку) конфигурирование компонентов приложений и управление жизненным циклом Java-объектов;
* Фреймворк [аспектно-ориентированного программирования](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5" \o "Аспектно-ориентированное программирование): работает с функциональностью, которая не может быть реализована возможностями [объектно-ориентированного программирования](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) на Java без потерь;
* Фреймворк доступа к данным: работает с [системами управления реляционными базами данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%A3%D0%A0%D0%91%D0%94) на Java-платформе, используя [JDBC](https://ru.wikipedia.org/wiki/JDBC)- и [ORM](https://ru.wikipedia.org/wiki/ORM)-средства и обеспечивая решения задач, которые повторяются в большом числе Java-based environments;
* Фреймворк управления [транзакциями](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)): координация различных [API](https://ru.wikipedia.org/wiki/API) управления транзакциями и инструментарий настраиваемого управления транзакциями для объектов Java;
* Фреймворк [MVC](https://ru.wikipedia.org/wiki/Model-view-controller): каркас, основанный на [HTTP](https://ru.wikipedia.org/wiki/HTTP) и [сервлетах](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%BB%D0%B5%D1%82" \o "Сервлет), предоставляющий множество возможностей для расширения и настройки (customization);
* Фреймворк [аутентификации](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F) и [авторизации](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F): конфигурируемый инструментарий процессов аутентификации и авторизации, поддерживающий много популярных и ставших индустриальными стандартами протоколов, инструментов, практик через дочерний проект [Spring Security](https://ru.wikipedia.org/wiki/Spring_Security).

### Hibernate[2]

Hibernate – фреймворк для языка программирования [Java](https://ru.wikipedia.org/wiki/Java), предназначенный для решения задач объектно-реляционного отображения ([ORM](https://ru.wikipedia.org/wiki/ORM)), самая популярная реализация спецификации [JPA](https://ru.wikipedia.org/wiki/Java_Persistence_API). Распространяется [свободно](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) на условиях [GNU Lesser General Public License](https://ru.wikipedia.org/wiki/GNU_Lesser_General_Public_License).

Позволяет сократить объёмы низкоуровневого программирования при работе с реляционными базами данных; может использоваться как в процессе проектирования системы классов и таблиц «с нуля», так и для работы с уже существующей [базой](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

Hibernate не только решает задачу связи классов Java с таблицами базы данных (и типов данных Java с типами данных [SQL](https://ru.wikipedia.org/wiki/SQL)), но и также предоставляет средства для автоматической генерации и обновления набора таблиц, построения запросов и обработки полученных данных и может значительно уменьшить время разработки, которое обычно тратится на ручное написание [SQL](https://ru.wikipedia.org/wiki/SQL)- и [JDBC](https://ru.wikipedia.org/wiki/JDBC)-кода. Hibernate автоматизирует генерацию SQL-запросов и освобождает разработчика от ручной обработки результирующего набора данных и преобразования объектов, максимально облегчая перенос (портирование) приложения на любые базы данных SQL.

Hibernate обеспечивает прозрачную поддержку сохранности данных (persistence) для «[POJO](https://ru.wikipedia.org/wiki/Plain_Old_Java_Object)» (то есть для стандартных Java-объектов); единственное строгое требование для сохраняемого класса — наличие [конструктора по умолчанию](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D0%BE%D1%80_%D0%BF%D0%BE_%D1%83%D0%BC%D0%BE%D0%BB%D1%87%D0%B0%D0%BD%D0%B8%D1%8E) (без параметров). Для корректного поведения в некоторых приложениях требуется также уделить внимание методам equals() и hashCode().

### Junit[3]

JUnit – фреймворк для [модульного тестирования](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) программного обеспечения на языке [Java](https://ru.wikipedia.org/wiki/Java).

Созданный [Кентом Беком](https://ru.wikipedia.org/wiki/%D0%9A%D0%B5%D0%BD%D1%82_%D0%91%D0%B5%D0%BA) и [Эриком Гаммой](https://ru.wikipedia.org/wiki/%D0%AD%D1%80%D0%B8%D0%BA_%D0%93%D0%B0%D0%BC%D0%BC%D0%B0), JUnit принадлежит семье фреймворков [xUnit](https://ru.wikipedia.org/wiki/XUnit" \o "XUnit) для разных языков программирования, берущей начало в [SUnit](https://ru.wikipedia.org/w/index.php?title=SUnit&action=edit&redlink=1" \o "SUnit (страница отсутствует)) Кента Бека для [Smalltalk](https://ru.wikipedia.org/wiki/Smalltalk" \o "Smalltalk). JUnit породил экосистему расширений – [JMock](https://ru.wikipedia.org/w/index.php?title=JMock&action=edit&redlink=1), [EasyMock](https://ru.wikipedia.org/w/index.php?title=EasyMock&action=edit&redlink=1" \o "EasyMock (страница отсутствует)), [DbUnit](https://ru.wikipedia.org/w/index.php?title=DbUnit&action=edit&redlink=1" \o "DbUnit (страница отсутствует)), [HttpUnit](https://ru.wikipedia.org/w/index.php?title=HttpUnit&action=edit&redlink=1" \o "HttpUnit (страница отсутствует)) и т.д.

Основные возможности и особенности JUnit:

* Простота и удобство использования. JUnit предоставляет простые и интуитивно понятные аннотации и методы для написания тестов;
* Автоматизация запуска тестов. JUnit позволяет легко автоматизировать запуск тестов, что важно для регулярного тестирования приложений;
* Встроенные утверждения. JUnit включает в себя набор встроенных утверждений(assertions), которые облегчают проверку ожидаемых результатов;
* Гибкость. JUnit поддерживает различные способы организации тестов: тесты можно группировать, запускать выборочно, фильтровать по тегам и т.д.;
* Интеграция с IDE. Большинство современных IDE, таких как Intelij IDEA, Eclipse, NetBeans, имеют встроенную поддержку JUnit, что упрощает разработку и запуск тестов;
* Расширяемость. JUnit можно расширять дополнительными библиотеками и фреймворками, например, Mockito для моков, AssertJ для более выразительных утверждений и т.д.

Использование JUnit позволяет разработчикам эффективно тестировать Java-код на уровне модулей, выявлять и исправлять ошибки на ранних этапах разработки, а также обеспечивать надежность и качество программного обеспечения.

### Apache Maven

Apache Maven – [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA) для [автоматизации сборки](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D1%81%D0%B1%D0%BE%D1%80%D0%BA%D0%B8) проектов на основе описания их структуры в файлах [POM](https://ru.wikipedia.org/w/index.php?title=POM_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%BE%D0%B2)&action=edit&redlink=1) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) [Project Object Model](https://en.wikipedia.org/wiki/Project_Object_Model)), на языке [XML](https://ru.wikipedia.org/wiki/XML). Проект Maven издаётся сообществом [Apache Software Foundation](https://ru.wikipedia.org/wiki/Apache_Software_Foundation), где формально является частью [Jakarta Project](https://ru.wikipedia.org/wiki/Apache_Jakarta_Project" \o ").

Maven обеспечивает [декларативную](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BA%D0%BB%D0%B0%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), а не [императивную](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) (в отличие от средства автоматизации сборки [Apache Ant](https://ru.wikipedia.org/wiki/Apache_Ant)) сборку проекта. В файлах описания проекта содержится его спецификация, а не отдельные команды выполнения. Все задачи по обработке файлов, описанные в спецификации, Maven выполняет посредством их обработки последовательностью встроенных и внешних плагинов.

Maven поддерживает принцип [соглашения по конфигурации](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B3%D0%BB%D0%B0%D1%88%D0%B5%D0%BD%D0%B8%D1%8F_%D0%BF%D0%BE_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8), заключающийся в том, что рассматриваемые аспекты нуждаются в конфигурации тогда и только тогда, когда этот аспект не удовлетворяет некоторой спецификации. Как следствие, это позволяет сократить количество требуемой конфигурации без потери гибкости.

Жизненный цикл maven-проекта — это список поименованных фаз, определяющий порядок действий при его построении. Жизненный цикл Maven содержит три независимых порядка выполнения:[5]

* clean – жизненный цикл для очистки проекта. Содержит следующие фазы:
  1. pre-clean
  2. clean
  3. post-clean
* default – основной жизненный цикл, содержащий следующие фазы:
  1. validate – выполняется проверка, является ли структура проекта полной и правильной.
  2. generate-sources
  3. process-sources
  4. generate-resources
  5. process-resources
  6. compile – компилируются исходные тексты.
  7. process-test-sources
  8. process-test-resources
  9. test-compile
  10. test – собранный код тестируется заранее подготовленным набором тестов.
  11. package – упаковка откомпилированных классов и прочих ресурсов. Например, в JAR-файл.
  12. integration-test – программное обеспечение в целом или его крупные модули подвергаются интеграционному тестированию. Проверяется взаимодействие между составными частями программного продукта.
  13. install – установка программного обеспечения в локальный Maven-репозиторий, чтобы сделать его доступным для других проектов текущего пользователя.
  14. deploy – стабильная версия программного обеспечения распространяется на удаленный Maven-репозиторий, чтобы сделать его доступным для других пользователей.
* site – жизненный цикл генерации проектной документации. Состоит из фаз:
  1. pre-site
  2. site
  3. post-site
  4. site-deploy

Стандартные жизненные циклы могут быть дополнены функционалом с помощью Maven-плагинов. Плагины позволяют вставлять в стандартный цикл новые шаги (например, распределение на сервер приложений) или расширять существующие шаги.

### Thymeleaf

Thymeleaf — это [шаблонизатор](https://en.wikipedia.org/wiki/Template_engine_(web)" \o "Шаблонизатор (веб))[Java](https://en.wikipedia.org/wiki/Java_(programming_language))[XML](https://en.wikipedia.org/wiki/XML) / [XHTML](https://en.wikipedia.org/wiki/XHTML) / [HTML5](https://en.wikipedia.org/wiki/HTML5) , который может работать как в веб-средах ( на основе [сервлетов](https://en.wikipedia.org/wiki/Servlet" \o "Сервлет) ), так и в не-веб-средах. Он лучше подходит для обслуживания XHTML/HTML5 на уровне представления веб-приложений на основе [MVC](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller), но может обрабатывать любые файлы XML даже в офлайн-средах. Он обеспечивает полную интеграцию [Spring Framework](https://en.wikipedia.org/wiki/Spring_Framework) .

В веб-приложениях Thymeleaf стремится стать полной заменой [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages" \o "Страницы JavaServer) (JSP) и реализует концепцию естественных шаблонов: файлы шаблонов, которые можно напрямую открывать в браузерах и которые по-прежнему корректно отображаются как веб-страницы.

Thymeleaf — это [программное обеспечение с открытым исходным кодом](https://en.wikipedia.org/wiki/Open-source_software), лицензированное по [лицензии Apache License 2.0](https://en.wikipedia.org/wiki/Apache_License_2.0).

Применение широкого спектра технологий и инструментов позволяет компании Innowise эффективно реализовывать проекты различного уровня сложности, обеспечивая высокое качество конечного продукта и удовлетворение потребностей клиентов из различных отраслей.

# выполнение индивидуального задания

## Индивидуальное задание

В рамках производственной практики студент должен разработать программное средство в рамках индивидуального задания от организации. Для разработки API магазина музыкальных инструментов необходимо реализовать следующие функциональные возможности: получение, добавление, обновление, удаление информации о музыкальных инструментах в ассортименте; получение информации о видах инструментов и получение инструментов по видам; складской учет; оформление, учет и обновление заказов; аутентификация и авторизация пользователей.

## Используемые технологии

* язык программирования: Java;
* фреймворк-основа: Spring Framework;
* ORM-фреймворк: Jpa/Hibernate;[6]
* аутентификация и авторизация: OAuth 2.0(keycloak);
* брокер сообщений: Apache Kafka;[1]
* обнаружение сервисов: Eureka;
* кеширование: Redis;
* СУБД: PostgreSQL;[4]
* документирование API: OpenAPI/Swagger.

## Архитектура проекта

Проект основан на микросервисной архитектуре и состоит из следующих микросервисов:

* config service: сервер конфигураций, то есть сервер, с которого остальные микросервисы читают конфигурации;
* discovery service: обнаружение сервисов, используется для регистрации остальных микросервисов для их последующего синхронного взаимодействия между собой и возможности реализации балансировки нагрузки на стороне клиента;
* gateway service: сервис-шлюз, предназначенный для создания единой точки взаимодействия с системой микросервисов;
* inventory service: сервис, ответственный за складской учет;
* order service: сервис, ответственный за заказы;
* product service: сервис, ответственный за товары;
* user service: сервис, ответственный за авторизацию и аутентификацию пользователей.

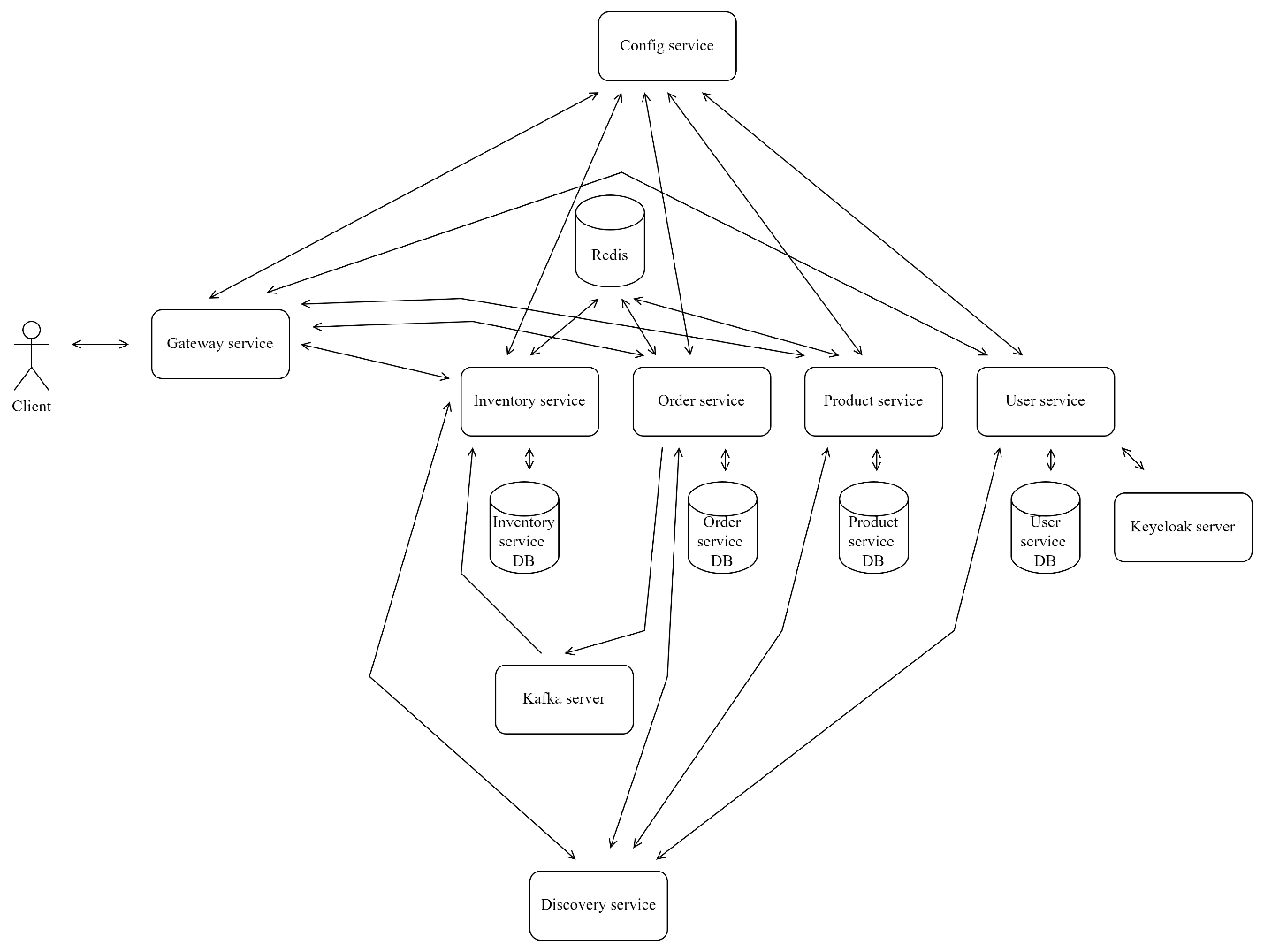


Рисунок 3 – Архитектура проекта

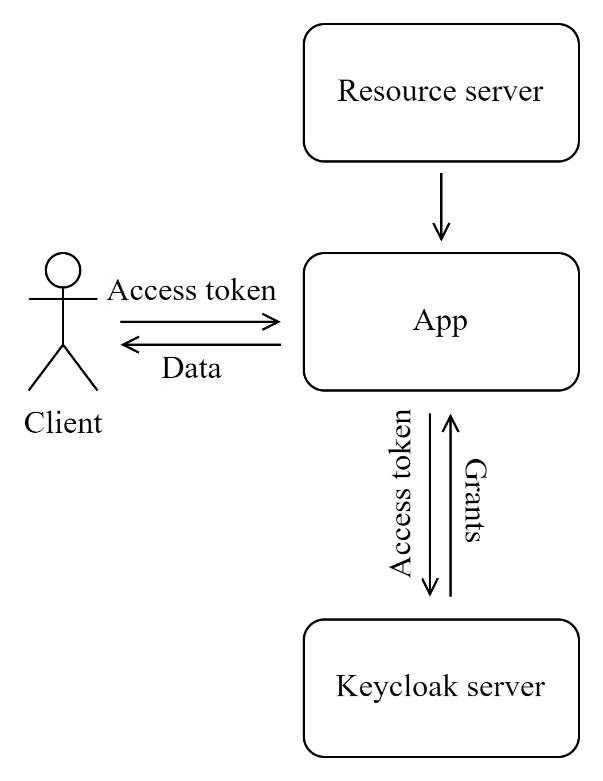


Рисунок 4 – Авторизация пользователя

## Функциональные требования

* присутствие аутентификации/авторизации пользователей с помощью JWT-токенов;
* получение, добавление, обновление, удаление информации о музыкальных инструментах в ассортименте;
* получение информации о видах инструментов и получение инструментов по видам;
* складской учет;
* оформление, учет и обновление заказов.

## Нефункциональные требования

* использование микросервисной архитектуры;
* OAuth 2.0 аутентификация/авторизация пользователей с использованием keycloak;
* использование механизма обнаружения служб;
* использование брокера сообщений;
* возможность легкого масштабирования системы.

Заключение

В ходе производственной практики приобретены знания о структуре организации, изучена нормативная документация и должностные инструкции. Практика также предоставила возможность получить опыт работы в команде разработчиков, изучить новые технологии и методы разработки, а также усовершенствовать навыки программирования в соответствии с современными тенденциями.

При выполнении задач на практике применялись теоретические знания, полученные в процессе обучения, что обеспечило эффективную работу в условиях ограниченных ресурсов и времени. Работа в команде позволила развить навыки эффективного взаимодействия с коллегами, обмена опытом и знаниями, а также принятия конструктивной критики. Это способствовало лучшему пониманию принципов Agile-разработки и важности коллективной работы для достижения общей цели.

Для выполнения индивидуального задания потребовалось изучение соответствующей литературы, а также использование опыта, полученного в ходе практики. Все задачи, поставленные перед началом практики, были успешно реализованы. Знания о современных технологиях и методах разработки программного обеспечения, полученные во время практики, будут полезны при выборе подходящих инструментов для написания дипломного проекта.

Приобретенный опыт поможет улучшить качество и структуру дипломной работы, а также повысить ее практическую значимость. Навыки программирования, анализа данных и проектирования систем, полученные в ходе практики, будут применены при разработке программного продукта и проведении исследований в рамках дипломного проекта.

Список использованных источников

[1] Серебряная Л.В. Структуры и алгоритмы обработки данных: учеб.-метод. пособие / Л.В. Серебряная, И.М. Марина. – Минск: БГУИР, 2013. – 5 с.

[2] JavaRush[Электронный ресурс]. – Режим доступа: <https://javarush.com/> – Дата обращения: 20.06.2024.

[3] JavaOnline[Электронный ресурс]. – Режим доступа: [https://java-online.ru](https://java-online.ru/)/ – Дата обращения: 21.06.2024.

[4] Ravesli[Электронный ресурс]. – Режим доступа: [https://ravesli.com/](https://ravesli.com/%20) – Дата обращения: 19.06.2024.

[5] StackOverflow[Электронный ресурс]. – Режим доступа: <https://stackoverflow.com/> – Дата обращения: 28.06.2024.

[6] Metanit[Электронный ресурс]. – Режим доступа: <https://metanit.com/cpp/> – Дата обращения: 17.06.2024.

[7] Innowise[Электронный ресурс]. – Режим доступа: <https://jobs-innowise.com/> – Дата обращения: 10.06.2024.

[8] Weeek[Электронный ресурс]. – Режим доступа: <https://weeek.net/> – Дата обращения: 15.06.2024.

[9] Atlassian[Электронный ресурс]. – Режим доступа: <https://www.atlassian.com/> – Дата обращения: 15.06.2024.

[10] SpringIo[Электронный ресурс]. – Режим доступа: <https://spring.io/> – Дата обращения: 13.06.2024.

Приложение А

Исходный код программы

@Service  
@RequiredArgsConstructor  
public class KeycloakServiceImpl  
 implements KeycloakService {  
  
 private final Keycloak keycloak;  
 private final RestTemplate restTemplate;  
  
 @Value("${keycloak.uri}")  
 private String keycloakURI;  
  
 @Value("${keycloak.realm}")  
 private String keycloakRealm;  
  
 @Value("${keycloak.clientId}")  
 private String keycloakClientId;  
  
 @Value("${keycloak.clientSecret}")  
 private String keycloakClientSecret;  
  
 @Override  
 public Optional<UserRepresentation> getUserByUsername(  
 String username) {  
 return keycloak.realm(keycloakRealm).users()  
 .search(username).stream()  
 .filter(u -> u.getUsername()  
 .equals(username)).findAny();  
 }  
  
 @Override  
 public UUID createUser(  
 UserRepresentation userRepresentation)  
 throws UserNotCreatedException {  
 keycloak.realm(keycloakRealm)  
 .users().create(userRepresentation);  
 return UUID.*fromString*(  
 getUserByUsername(userRepresentation  
 .getUsername()).  
 orElseThrow(() ->  
 new UserNotCreatedException(  
 Messages.*USER\_NOT\_CREATED*)).getId());  
 }  
  
 @Override  
 public void assignRealmRolesToUser(UUID userId,  
 List<RoleRepresentation> roles) {  
 keycloak.realm(keycloakRealm).  
 users().get(userId.toString())  
 .roles().realmLevel().add(roles);  
 }  
  
 @Override  
 public Optional<RoleRepresentation> getRealmRoleRepresentation(  
 String name) {  
 return keycloak  
 .realm(keycloakRealm).  
 roles().list().stream().filter(r ->  
 r.getName().equals(name)).findAny();  
 }  
  
 @Override  
 public Optional<RefreshAndAccessTokenDto> getUserTokensByRefreshToken(  
 String refreshToken) {  
 //Desired functionality is not supported  
 //by java keycloak client,  
 //so use RestTemplate here  
 HttpHeaders headers = new HttpHeaders();  
 headers.setContentType(  
 MediaType.*APPLICATION\_FORM\_URLENCODED*);  
 MultiValueMap<String, String> map =  
 new LinkedMultiValueMap<>();  
 map.add("grant\_type", "refresh\_token");  
 map.add("refresh\_token", refreshToken);  
 map.add("client\_id", keycloakClientId);  
 map.add("client\_secret", keycloakClientSecret);  
 HttpEntity<MultiValueMap<String, String>> entity =  
 new HttpEntity<>(map, headers);  
 try {  
 return Optional.*ofNullable*(restTemplate  
 .postForObject(keycloakURI + "/realms/"  
 + keycloakRealm +  
 "/protocol/openid-connect/token",  
 entity, RefreshAndAccessTokenDto.class));  
 } catch (HttpClientErrorException e) {  
 return Optional.*empty*();  
 }  
 }  
  
 @Override  
 public Optional<RefreshAndAccessTokenDto> getUserTokensByCredentials(  
 String username, String password) {  
 TokenManager tokenManager = Keycloak  
 .*getInstance*(keycloakURI,  
 keycloakRealm, username, password,  
 keycloakClientId, keycloakClientSecret)  
 .tokenManager();  
 try {  
 String accessToken =

tokenManager.getAccessTokenString();  
 String refreshToken = tokenManager.refreshToken()  
 .getRefreshToken();  
 return Optional.*ofNullable*(RefreshAndAccessTokenDto  
 .*builder*()  
 .refreshToken(refreshToken)  
 .accessToken(accessToken).build());  
 } catch (NotAuthorizedException e) {  
 return Optional.*empty*();  
 }  
 }  
  
 @Override  
 public void logout(UUID userId) {  
 keycloak.realm(keycloakRealm)

.users().get(userId.toString())

.logout();  
 }  
}

# Формирование технической задачи дипломного проекта

## Определение целей и задач проекта

### Цель проекта

Разработать игровое приложение по коллекционированию игровых существ, состоящее из серверной части, написанной на языке Java, и клиентской части, реализованной с помощью игрового движка Godot.

### Задачи проекта

* создание удобного и интуитивно понятного интерфейса;
* разделение серверной и клиентской части;
* разработка алгоритов взаимодействия с пользователем и отправки запросов на сервер в клиентской части;
* разработка алгоритмов обработки запросов и реализации игровой логики в серверной части;
* разработка игровых механик и функционала приложения.

## Функциональные требования

### Основные модули

* клиентский модуль: взаимодействие с пользователем, предоставление удобного и интуитивно понятного интерфейса, создание приятной визуальной составляющей;
* серверный модуль: обработка запросов от клиента, реализация логики по поимке, содержанию игровых существ, по организации сражений существ друг с другом, а также управление аккаунтами пользователей.

### Пользовательские роли

* обычный пользователь: лицо, взаимодействующее с приложением и участвующее в игровом процессе.

## Нефункциональные требования

### Производительность

* приложение должно быстро обрабатывать запросы, так, чтобы это было незаметно для конечного пользователя;
* приложение должно эффективно обрабатывать большие объемы данных.

### Надёжность

* масштабируемость – приложение должно легко масштабироваться и адаптироваться к растущим нагрузкам при появлении новых пользователей;
* приложение должно обеспечивать отказоустойчивость и свою бесперебойную работу;
* должно быть обеспечено регулярное резервное копирование данных.

### Безопасность

* защита пользовательских данных с использованием шифрования;
* аутентификация и авторизация пользователей.

### Интерфейс

* удобный и интуитивно понятный интерфейс;
* приятная визуальная составляющая;
* поддержка различных разрешений экрана (адаптивный дизайн).