# Отчёт по практике

# Введение

**Цель**: создать экосистему студенческого общежития с обратной связью и лентой новостей

**Задачи**:

* Определить функции приложения, которые необходимы для комфортного пользования.
* Выбрать предполагаемый стек технологий и используемых инструментов.
* Создать макет интерфейса приложения «Экосистема студенческого общежития», позволяющий реализовать весь функционал, при этом сохраняющий удобство пользования.
* Описать принцип работы приложения, работу всех используемых инструментов.
* Составить этапы тестирования макетов и функций приложения, а также возможных проблемных частей в приложении.
* Обобщить результаты проделанной работы и сделать выводы по теме исследования.

# Аналитика

Один из важных этапов проектирования приложения – проведение исследований. Прежде всего необходимо определить, решение каких задач может потребоваться потенциальному пользователю. После этого необходимо выделить задачи и процессы, которые будут реализованы в проекте. Далее необходимо проработать взаимодействия между пользователями и сценарии по работе с приложением.

Для этого были составлены следующие диаграммы:

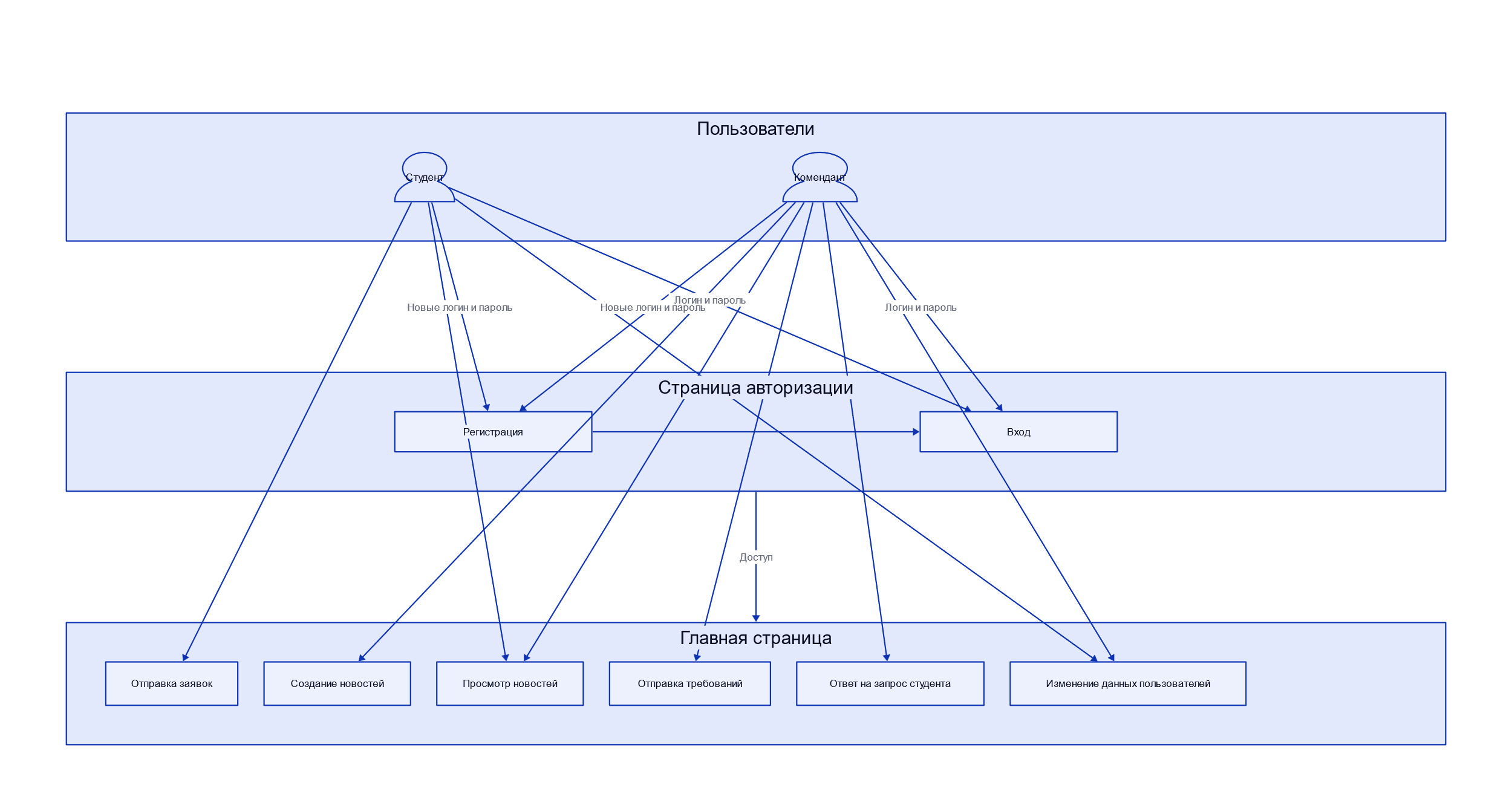


Рис.1 Use Case Diagram

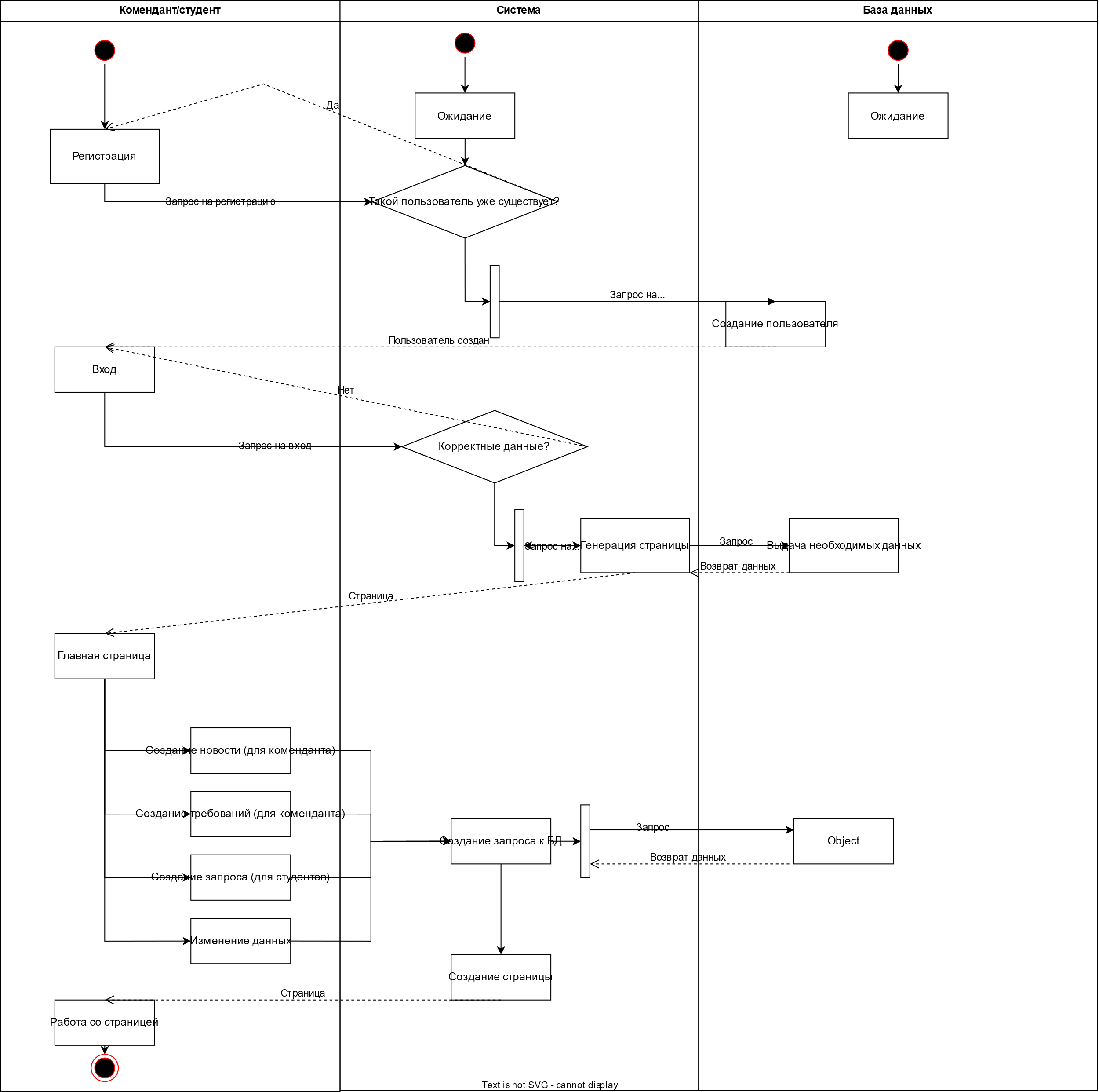


Рис.2 Activity Diagram

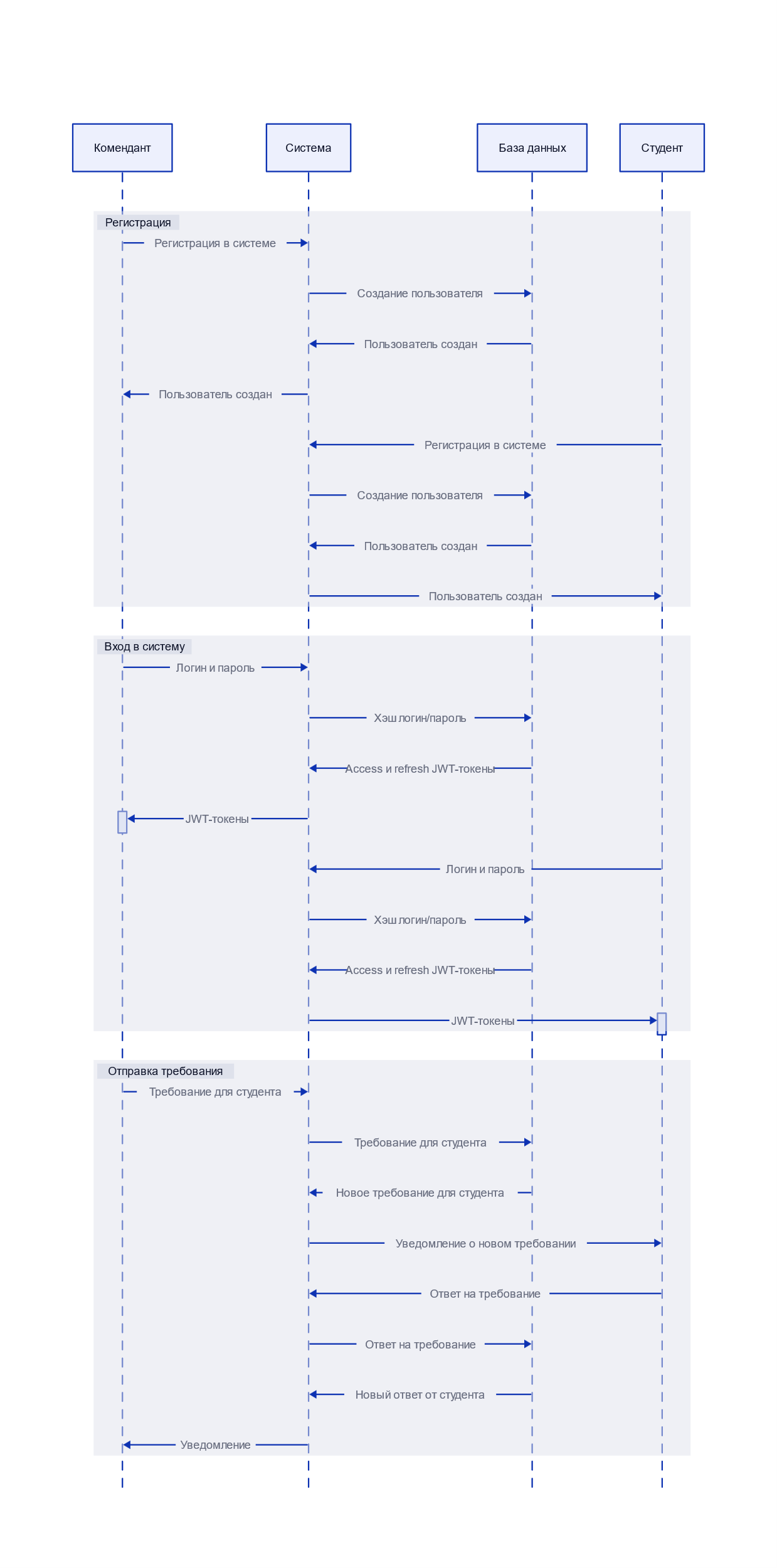
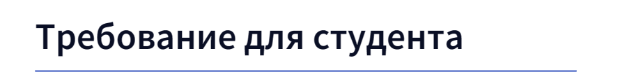


Рис.3 Sequences Diagram для пользователя

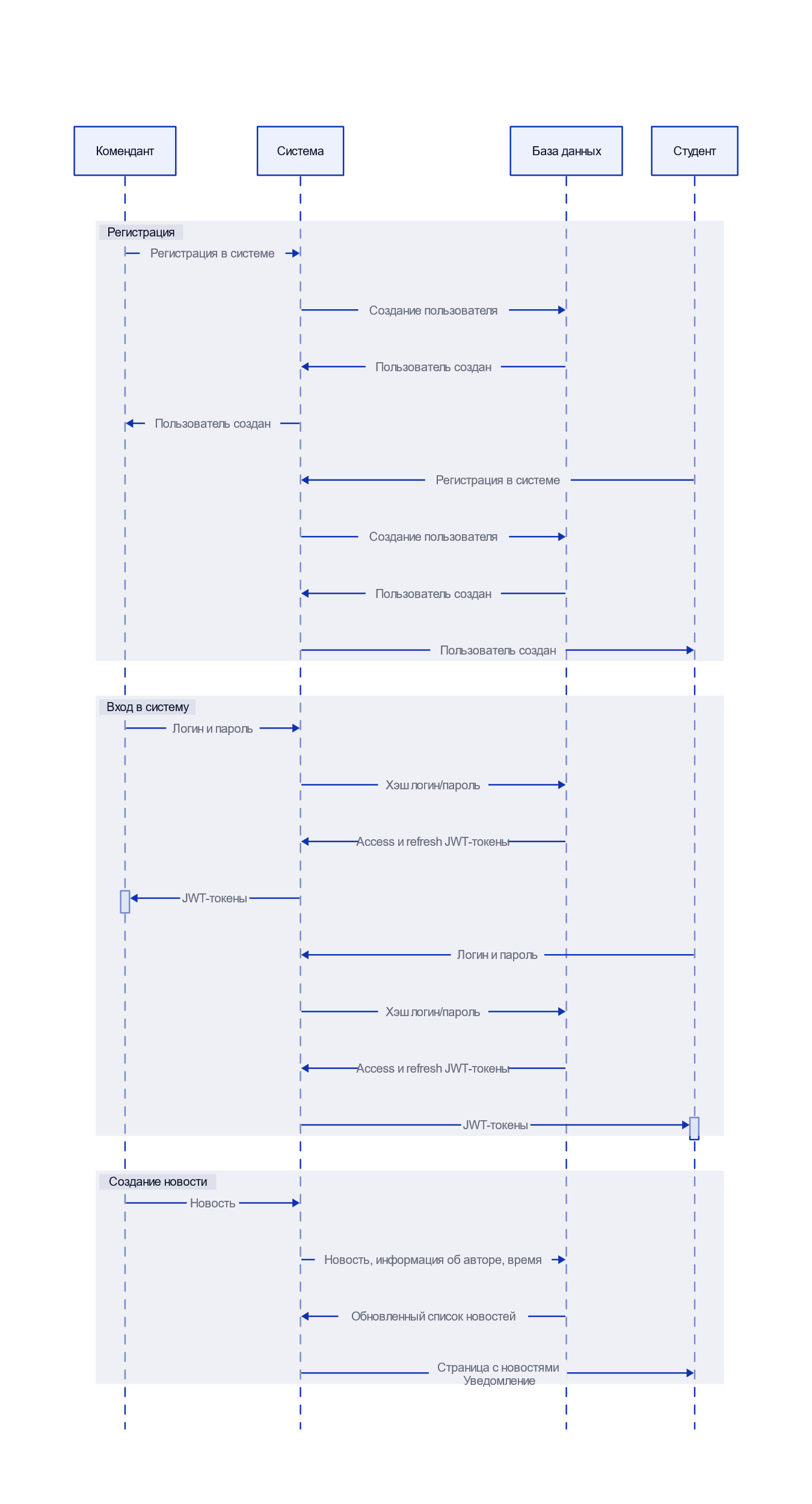


Рис.4 Sequences Diagram для создания новости

**User stories**

***Студент:***

Как студент, я хочу просматривать новости и объявления, чтобы быть в курсе последних событий и важной информации об общежитии.

Как студент, я хочу иметь возможность отправлять запросы или задавать вопросы через приложение, чтобы быстро связываться с комендантами по вопросам, которые могут возникнуть.

Как студент, я хочу получать уведомления от коменданта, чтобы быть в курсе важных сообщений или требований, например, о сдаче документов или оплате.

Как студент, я хочу иметь возможность отвечать на требования и сообщения коменданта, чтобы подтверждать выполнение требований или задавать уточняющие вопросы.

***Для коменданта:***

Как комендант, я хочу создавать новости и объявления через приложение, чтобы информировать студентов об актуальных событиях и требованиях.

Как комендант, я хочу отправлять информацию, оповещения или требования студентам, указывая ФИО или номер комнаты, чтобы оперативно общаться с ними.

Как комендант, я хочу видеть запросы от студентов и иметь возможность отвечать на них через приложение, чтобы облегчить коммуникацию и решение вопросов.

Как комендант, я хочу отслеживать статус запросов от студентов, чтобы знать, какие запросы были выполнены или ожидают решения.

На основе полученных данных можно начать проектирование продукта.

## Выбор стека для мобильного приложения

Чтобы выбрать подходящий набор технологий для мобильного приложения, необходимо учитывать масштаб бизнеса и потребности решения. Для MVP стартапа лучше всего подходит кроссплатформенный метод. Для уже успешного бизнеса лучшим выбором может стать разработка нативных приложений, а для первого немасштабируемого MVP начинающего стартапа подойдет гибридная модель.

Исходя из этого, стоит рассмотреть наиболее популярные кросс-платформенные фреймворки для разработки на android, ios и windows.

**Flutter**

Flutter – это SDK от Google, созданный для платформ:

* Android
* Google Fuchsia
* iOS и Mac
* Linux
* Windows
* Веб-разработка

Таким образом, включив Flutter в свой стек технологий мобильной разработки, вы сможете использовать один код для любой из этих платформ. Это также оптимальный выбор, если вы хотите сократить время выхода на рынок, поскольку процесс сборки происходит быстро и надежно, а также исправление ошибок и обнаружение новых функций.

**Xamarin**

Платформа, разработанная Microsoft, предлагает нативные приложения для:

* Android
* iOS
* Windows

Фреймворк использует .NET и C#, при этом 96% его кода является многоразовым. Он допускает как MVVM, так и MVC дизайн и удобен для отладки. Кроме того, он активно поддерживается сообществом, что означает, что это стабильная современная платформа.

**React Native**

React Native– это JavaScript-фреймворк, используемый для создания решений для:

* Android
* iOS
* Web
* Universal Windows Platform (UWP)

Почему мы выбираем **Flutter**:

1. С Flutter можно выпускать приложения для разных версий Android и iOS без дополнительных телодвижений: программы спокойно запускаются даже на таких старых версиях, как Android Jelly Bean и iOS 8.
2. При прочих равных можно сказать, что Flutter превосходит конкурентов и демонстрирует самую высокую производительность благодаря современному языку Dart и собственному движку рендеринга.
3. Flutter стал более популярен за последние годы, в отличие от React Native и Xamarin, хотя является относительно молодым фреймворком.

# Обзор backend части приложения

# Стек технологий

**Dart**

Язык программирования Dart — это язык, разработанный компанией Google, который широко используется для создания приложений на платформе Flutter. Он является статически типизированным, объектно-ориентированным и компилируемым в машинный код или в JavaScript.

***Преимущества:***

• Одно из главных преимуществ Dart заключается в его эффективной производительности. Dart компилируется в машинный код, что делает его быстрым и эффективным при выполнении приложений. Кроме того, в Dart присутствует сборщик мусора, который автоматически освобождает память от неиспользуемых объектов, упрощая процесс управления памятью.

• Другим преимуществом Dart является его универсальность. Он может быть использован для создания различных типов приложений, включая мобильные приложения, веб-приложения и даже серверные приложения. Это позволяет разработчикам использовать один язык программирования для создания приложений на разных платформах.

• Также, Dart предлагает современные возможности языка программирования, такие как асинхронное программирование и поддержку потоков, а также множество библиотек и инструментов для разработки приложений.

**Firebase Firestore**

Одним из основных компонентов Firebase является Firestore - облачная база данных, которая предлагает гибкую структуру для хранения и синхронизации данных в режиме реального времени.

***Основные преимущества Firebase Firestore:***

• Удобство использования: Firestore предлагает простой и интуитивно понятный API, что упрощает работу с базой данных и снижает время разработки.

• Режим реального времени: Firestore поддерживает синхронизацию данных между клиентами в режиме реального времени. Это означает, что любые изменения, внесенные в базу данных, мгновенно отражаются на всех подключенных устройствах.

• Горизонтальное масштабирование: Firestore автоматически управляет горизонтальным масштабированием вашей базы данных. Это означает, что вам не нужно беспокоиться о производительности или масштабируемости ваших приложений.

• Аутентификация пользователей: Firebase предоставляет удобные инструменты для аутентификации пользователей, включая поддержку социальных сетей и двухфакторную аутентификацию.

• Безопасность: Firestore предлагает мощные средства для защиты данных, включая правила безопасности на уровне документа или коллекции. Это позволяет определить, кто может просматривать и изменять ваши данные.

**Firebase Cloud Messaging (FCM)**

Firebase Cloud Messaging (FCM) — это кросс-платформенный сервис обмена сообщениями, который позволяет разработчикам отправлять уведомления, а также обменяться данными с клиентскими приложениями на различных платформах, включая Android, iOS и веб-приложения. FCM обеспечивает надежную доставку сообщений с использованием push-уведомлений и обслуживает миллионы устройств одновременно.

***Преимущества:***

• Надежная доставка: FCM использует оптимизированные каналы связи для доставки сообщений клиентскому приложению. Он обеспечивает гарантированную доставку сообщений даже в случае временной недоступности устройства.

• Масштабируемость: FCM поддерживает обслуживание большого числа клиентских приложений и устройств одновременно. Это позволяет разработчикам масштабировать свои приложения, не беспокоясь о проблемах с пропускной способностью или надежностью доставки.

• Гибкость: FCM позволяет отправлять не только уведомления, но и обмениваться данными между клиентскими приложениями и сервером. Разработчики могут легко обновлять и расширять свои приложения, используя эту функциональность.

• Собственный панель инструментов: Firebase предоставляет панель инструментов, которая позволяет разработчикам легко настраивать и отслеживать свои уведомления и данные, отправленные через FCM.

# Описание принципа работы бекэнд части

1. Аутентификация и авторизация пользователей: Пользователи имеют возможность зарегистрироваться и войти в приложение, используя свой логин и пароль. После успешной аутентификации, пользователь получает уникальный токен доступа, который используется для авторизации запросов к API приложения.

2. Создание и управление новостями и объявлениями: Коменданты или студсовет могут создавать новости или объявления через интерфейс приложения. Имеют возможность добавить текст, заголовки, изображения и другие медиафайлы если это необходимо. Для эффективного управления новостями и объявлениями используется база данных.

3. Обратная связь: чтобы обеспечить коммуникацию между комендантами и студентами, необходимо предоставить функциональность обратной связи. Комендант может отправить информацию, оповещение или требование студенту, указав его ФИО или номер комнаты. Студенты могут отвечать на эти сообщения и отправлять запросы или задавать вопросы через интерфейс приложения.

4. Управление запросами: Система предоставляет механизм для управления запросами от студентов. Коменданты имеют возможность видеть и отвечать на запросы, отслеживать их статус и отмечать, когда запрос был выполнен или закрыт.

5. Обработка данных: важно обеспечить безопасное и эффективное хранение и обработку данных приложения. Это включает в себя использование базы данных для хранения пользовательских данных, новостей, объявлений и запросов, а также соответствующие механизмы шифрования и защиты данных пользователей.

**Что такое тестирование?**

**Тестирование программного обеспечения (Software Testing) —** проверка соответствия реальных и ожидаемых результатов поведения программы, проводимая на конечном наборе тестов, выбранном определённым образом.

**В чем цель тестирования?**

**Цель тестирования —** проверка соответствия ПО предъявляемым требованиям, обеспечение уверенности в качестве ПО, поиск очевидных ошибок в программном обеспечении, которые должны быть выявлены до того, как их обнаружат пользователи программы.

**Для чего проводится тестирование ПО?**

* Для проверки соответствия требованиям.
* Для обнаружения проблем на более ранних этапах разработки для предотвращения повышения стоимости продукта.
* Обнаружение вариантов использования, которые не были предусмотрены при разработке.

**Современные способы тестирования.**

Основные способы тестирования: ручное и автоматическое. Наиболее оптимальным подходом будет комбинирование этих тестов.

1. **Ручное тестирование** — это метод, при котором тестировщики вручную проверяют продукт на соответствие требованиям и оценивают его работоспособность. В процессе ручного тестирования тестировщик взаимодействует с интерфейсом, выполняет определенные действия и проверяет, что результаты соответствуют ожиданиям. Ручное тестирование позволяет проверить различные сценарии использования, а также оценить визуальный дизайн и удобность пользования.
2. **Автоматизированное тестирование** — это использование специальных инструментов и написание скриптов, которые выполняют определенные тестовые сценарии автоматически. Тестовые скрипты могут симулировать действия пользователя, с помощью автоматизированного тестирования можно ускорить процесс тестирования, снизить вероятность человеческой ошибки и повысить надежность результатов.

После тестирования и вывода продукта на рынок необходима обратная связь от пользователей.

**Зачем нужна обратная связь?**

Сбор обратной связи и анализ отзывов и на основе полученной информации помогают удерживать старых и привлекать новых пользователей.

Существует несколько способов получения и интерпретации обратной связи.

Они могут быть разными в зависимости от контекста и цели обратной связи.

1. **Анкеты и опросы**: Этот метод включает создание структурированных вопросов и их распространение среди целевой аудитории. Анкеты могут быть распространены как в печатном формате, так и в электронном виде через электронную почту, социальные сети или специализированные платформы для проведения опросов. Анкеты могут быть как открытыми (с возможностью свободного ответа), так и закрытыми (с предложенными вариантами ответов). После получения ответов, результаты анкеты анализируются для выявления общих тенденций и взглядов аудитории.
2. **Фокус-группы**: Этот метод включает организацию групповых дискуссий с небольшой группой представителей целевой аудитории. Участники фокус-группы могут обсуждать определенную тему, задавать вопросы друг другу и делиться своими мнениями и отзывами. Фокус-группы могут проводиться лично, по телефону или онлайн через видеоконференции. Результаты дискуссий могут быть записаны и проанализированы для получения ценных выводов.
3. **Открытые форумы:** Организация открытых форумов или платформ, где люди могут свободно высказывать свое мнение и делиться отзывами и идеями. Это может быть веб-сайт, специальный онлайн-форум, чат-бот или социальная сеть. Открытые форумы позволяют людям общаться между собой, задавать вопросы, отвечать на них и обмениваться идеями. Результаты открытых форумов могут быть анализированы для выявления общих мнений и предпочтений.
4. **Обратная связь через социальные сети:** Отслеживание и анализ отзывов и комментариев, оставленных людьми на платформах социальных сетей, таких как Вконтакте, Telegram, Twitter и др. Мнения и отзывы, оставленные публично, могут быть анализированы для определения общего мнения пользователей о продукте, бренде или услуге. Это также может помочь в выявлении проблем и сильных сторон продукта или услуги.
5. **Клиентские обзоры и оценки:** Изучение обзоров, оставленных клиентами на различных платформах, таких как Google, Yelp, Tripadvisor и другие. Клиенты имеют возможность оставить отзывы и оценки после использования продукта или услуги. Это дает возможность получить представление о том, что клиенты думают о вашей компании или продукте.

Интерпретация обратной связи требует анализа полученных данных и выделения ключевых выводов. Важно обратить внимание не только на конкретные мнения, но и на общие тенденции и паттерны. Кроме того, автоматизация обработки обратной связи с помощью машинного обучения и анализа данных может существенно упростить этот процесс.

**Тестирование**

1. Проверка работоспособности авторизации всех ролей (Комендант, студсовет, жилец);
2. Сравнить возможности каждой роли с предполагаемыми;
3. Тестирование менеджмента новостей и объявлений для комендантов и студсовета;
4. Проверка отображения новостей в ленте;
5. Тестирование системы связи Комендант-Студент, в том числе обратной связи;
6. Тестирование запросов жильцов к комендантам, в том числе возможности взаимодействия комендантов с запросами;

Журнал тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Задача | Шаги | Ожидаемый результат | Полученный результат |
| 1 | Авторизация по логину и паролю. | Нажать кнопку авторизация  Ввести логин;  Ввести пароль;  Нажать кнопку «отправить»; | Ответ на запрос пришел с кодом 200, пользователь авторизован, можно зайти в личный кабинет | Ответ на запрос пришел с кодом 200, пользователь авторизован, можно зайти в личный кабинет |
| 2 | Регистрация | Нажать кнопку регистрация;  Ввести логин, пароль, email, номер комнаты;  Подтвердить email и номер комнаты;  Авторизоваться; |  |  |
| 2 | Вход в личный кабинет | Авторизоваться;  Нажать на кнопку ЛК; | Открывается личный кабинет | Открывается личный кабинет |
| 3 | Просмотр отображаемой после авторизации роли. | Войти в ЛК;  Проверить показанную роль; | В личном кабинете отображается нужная роль, а также возможности только этой роли | В личном кабинете отображается нужная роль, а также возможности только этой роли |
| 4 | Опубликовать новость для ролей Комендант и Студсовет. | Войти в ЛК; Нажать кнопку написать новость;  Ввести содержание;  Добавить тэги;  Указать уровень доступа  Нажать кнопку опубликовать новость | Новость отображается в ленте у указанных пользователей | Новость отображается в ленте у указанных пользователей |
| 5 | Отправить сообщение жильцу от имени Коменданта | Войти в ЛК;  Нажать кнопку отправить сообщение;  Выбрать комнату;  Выбрать жильца;  Ввести сообщение;  При необходимости прикрепить файлы;  Нажать кнопку отправить |  |  |
| 6 | Отправить обратную связь на сообщение | Войти в ЛК;  Открыть панель сообщения;  Выбрать нужное сообщение;  Нажать кнопку ответить;  Написать ответ;  Прикрепить файлы при необходимости;  Нажать кнопку отправить; |  |  |
| 7 | Отправить запрос к руководству  Для жильцов | Войти в ЛК;  Открыть панель запросы;  Нажать кнопку новый запрос;  Написать содержание запроса;  Прикрепить файлы при необходимости;  Нажать кнопку отправить4 |  |  |
| 8 | Проверка возможности обработать запрос |  |  |  |

# Заключение

В ходе практической работы студенты приобрели навыки и знания в области создания диаграмм и проектировании идеи приложения, изучили инструменты, которые для этого принято использовать. Также был составлен набор тестов для приложения и возможные пользовательские сценарии.

Был выбран стек технологий для создания приложения dormitory-ecosystem: Flutter, Dart и PostgreSQL.

Ссылка на github: https://github.com/vedrent/dormitory-ecosystem

# Список использованных источников

1. "Руководство по технологическому стеку разработки мобильных приложений на 2023 год" // Notissimus URL: https://notissimus.com/biznes-rukovodstvo-po-tehnologicheskomu-steku-razrabotki-mobilnyh-prilozhenij-na-2023-god/ (дата обращения: 08.10.2023).
2. "Cross-Platform Frameworks for Mobile Development" // Tproger URL: https://tproger.ru/translations/cross-platform-frameworks-for-mobile-development (дата обращения: 08.10.2023).
3. "Гибкая методология управления проектами: Agile vs. Traditional" // Habr URL: https://habr.com/ru/articles/447640 (дата обращения: 03.10.2023).
4. "Как тестировать веб-сервисы: советы тестировщика Wrike" // Habr URL: https://habr.com/ru/companies/wrike/articles/343988 (дата обращения: 03.10.2023).
5. Официальный сайт Dart // URL: https://dart.dev (дата обращения: 03.10.2023).
6. "Как мы разрабатываем клиент-серверное приложение в DOM.RU: опыт и практические советы" // Habr URL: https://habr.com/ru/companies/domclick/articles/510354 (дата обращения: 03.10.2023).
7. "Как выбрать багтреккер: сравнение Jira, Redmine и YouTrack" // Habr URL: https://habr.com/ru/articles/549054 (дата обращения: 10.10.2023).
8. "Виды и типы тестирования: обзор основных методов" // TestEngineer URL: https://testengineer.ru/vidy-tipy-testirovaniya (дата обращения: 10.10.2023).
9. "Feedback (Обратная связь): как её правильно собирать и обрабатывать" // Academy of Capital URL: https://academy-of-capital.ru/blog/fidbek (дата обращения: 10.10.2023).