

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

МО и ПЭВМ

Направление подготовки

09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему

**Подсистема приложения дополненной
реальности «Живая клетка»**

Студент

Ефакина Евгения Сергеевна

(ФИО полностью)

Руководитель

Кольчугина Е.А.

(фамилия, инициалы)

Нормоконтролёр

Такташкин Д.В.

(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой

Макарычев П.П.

(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК

Попова Н.А.

(фамилия, инициалы)

Пенза, 2020

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет

Вычислительной техники

Кафедра

МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ П.П. Макарычев
«__» _____ 20__ г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Ефакина Евгения Сергеевна гр. 16ВИ1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Подсистема приложения дополненной реальности «Живая клетка»

Тема утверждена приказом ПГУ № _____ от "_____" _____ 20__

2. Руководитель работы д.т.н., профессор каф. «МО и ПЭВМ» Кольчугина Е.А.

3. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать подсистему приложения дополненной реальности, моделирующее строение живой клетки и ее органелл, а также процессы, происходящие в клетке. Приложение должно быть включено в образовательную систему Experiment, через которую происходит авторизация пользователя.

Приложение должно работать на устройствах с операционной системой Android выше седьмой версии и имеющих поддержку технологии дополненной реальности.

Для разработки использовать среду Android Studio и редактор 3D графики Blender.

4. Перечень подлежащих разработке вопросов

1) Анализ требований и постановка задачи на разработку подсистемы приложения дополненной реальности «Живая клетка»

2) Проектирование приложения «Живая клетка»

3) Структура данных для хранения информации о клетках и органеллах

4) Структура программы для отображения в AR моделей клеток

5) Реализация логики работы подсистемы приложения «Живая

клетка»

6)	Тестирование	подсистемы	приложения	«Живая
----	--------------	------------	------------	--------

клетка»

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ требований и постановка задачи	10%	13.04.20 – 20.04.20	
2.	Проектирование приложения дополненной реальности «Живая клетка»	20%	20.04.20 – 04.05.20	
3.	Разработка подсистемы приложения дополненной реальности «Живая клетка»	50%	04.05.20 – 27.05.20	
4.	Тестирование подсистемы приложения «Живая клетка»	10%	27.05.20 – 08.06.20	
5.	Оформление пояснительной записки	10%	08.06.20 – 16.06.20	

Дата выдачи задания «30» декабря 2019 г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____ «30» декабря 2019 г.
(подпись, дата)

Работу к защите допустить
Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 88 листов, 45 рисунков, 20 таблиц, 15 использованных источников и 4 приложения.

ДОПОЛНЕННАЯ РЕАЛЬНОСТЬ, ANDROID STUDIO, BLENDER, ИНТЕРФЕЙС, ИЗУЧЕНИЕ КЛЕТОК, JAVA.

Объект исследования: подсистема приложения дополненной реальности «Живая клетка».

Цель работы: разработать клиентскую часть приложения, которое будет использоваться в образовательных учреждениях для изучения строения клеток и изучения биологических процессов, происходящих в них.

Средства разработки: Android Studio, Java, Blender.

Результаты работы: спроектирована и разработана подсистема приложения дополненной реальности «Живая клетка».

Область применения: средние общеобразовательные учреждения.

					ПГУ 09.03.02– 08БР161.12 ПЗ								
Изм.	Лист	№ докум.	Подпись	Дата	Подсистема приложения дополненной реальности «Живая клетка».			Лист		Лист		Листов	
Разраб.		Ефакина Е.С.								3		88	
Провер.		Кольчугина Е.А.											
Реценз.								гр. 16ВИ1					
Н. Контр.		Такташкин Д.В.											
Утверд.		Макарычев П.П.											

Содержание

1	Анализ требований и постановка задачи	8
1.1	Анализ программных продуктов аналогичного назначения.....	8
1.2	Постановка задачи на разработку подсистемы приложения «Живая клетка»	14
1.3	Анализ функциональных требований	16
1.4	Выбор средств разработки	20
2	Проектирование подсистемы приложения дополненной реальности «Живая клетка»	22
2.1	Функциональное моделирование приложения «Живая клетка»	22
2.2	Проектирование структур данных	26
2.2.1	Логическая модель базы данных	27
2.2.2	Физическая модель базы данных	28
2.3	Проектирование программных средств разрабатываемого приложения.....	29
2.3.1	Диаграмма классов приложения «Живая клетка»	29
2.3.2	Диаграмма состояний для класса Task приложения «Живая клетка»...	32
2.3.3	Диаграмма деятельности процесса разработки анимации процессов, происходящих в клетке	33
2.3.4	Диаграмма последовательности процесса разработки анимации процессов, происходящих в клетке	35
2.3.5	Диаграмма кооперации для процесса создания моделей клеток	37
2.4	Проектирование интерфейса приложения «Живая клетка»	38
3	Реализация подсистемы приложения дополненной реальности «Живая клетка»	43
3.1	Реализация пользовательского интерфейса	43
3.2	Реализация программных средств	45
3.2.1	Диаграмма компонентов подсистемы приложения «Живая клетка»	45
3.2.2	Представление развертывания приложения «Живая клетка».....	48
3.2.3	Реализация процесса загрузки моделей клеток	49
4	Тестирование подсистемы приложения дополненной реальности «Живая клетка»	52
4.1	Структурное тестирование	52
4.1.1	Метод покрытия операторов	53

4.1.2	Метод покрытия условий.....	54
4.1.3	Метод покрытия маршрутов.....	55
4.2	Анализ качества кода	56
4.2.1	Метрики кода.....	56
4.2.2	SLOC-метрика	56
4.2.3	Цикломатическая сложность Маккейба.....	57
4.2.4	Метрики Холстеда.....	58
5	Планирование и расчет бюджета разработки подсистемы приложения дополненной реальности «Живая клетка»	61
	Заключение	68
	Список использованных источников	69
	Приложение А. Глоссарий.....	71
	Приложение Б. Листинг программы на языке Java	73
	Приложение В. Результаты структурного тестирования	83
	Приложение Г. Акт о результатах внедрения.....	86

Введение

В настоящее время информационные технологии развиваются все более стремительно, что ведет к внедрению цифровых технологий в различные сферы общества. Каждый день люди пользуются техническими устройствами, которые облегчают выполнение бытовых задач. Любой взрослый человек и почти каждый школьник имеет в своем распоряжении смартфон или планшет. Зачастую, школьники проводят слишком много времени за своими смартфонами и отвлекаются на них в процессе обучения. Так как телефон является неотъемлемой частью современных школьников, следовало бы вывести образовательный процесс на новый уровень, сделать его более интересным и увлекательным, чем традиционное обучение [1]. Внедрение современных технологий позволило бы заинтересовать учеников и сделать обучение наиболее приближенным к их досугу.

Одной из технологий, набирающих все большую популярность, является технология дополненной реальности (AR/ augmented reality), которая позволяет совмещать реальный мир с виртуальным. Данная технология уже используется и в сфере образования [1]. С помощью AR-технологий можно визуализировать то, что нельзя визуализировать с помощью учебников и слов учителей. Гораздо интереснее рассматривать объемную модель изучаемого объекта, чем 2D-иллюстрацию в книге.

В школьном курсе предметов, есть множество дисциплин, в которых бы использование технологии дополненной реальности помогло бы увеличить интерес и вовлеченность школьников к данному предмету. Одним из таких предметов является биология. Биология изучает строение живых организмов и процессы жизнедеятельности этих организмов. Также, существуют разделы, изучающие микромир: строение клеток, вирусов, бактерий. Микромир – это мир, который человек не сможет увидеть без помощи специальных устройств и который сложно представить. Именно поэтому существует необходимость в создании образовательных приложений, позволяющих рассмотреть и изучить микроэлементы с помощью объемных моделей.

Данная работа направлена на создание клиентской части приложения дополненной реальности «Живая клетка», которое бы позволило изучить животные и растительные клетки, вирусы и бактерии, а также процессы, происходящие в них.

Выполненная работа позволит увеличить заинтересованность школьников к данной теме, позволит проверить знания, полученные по данному разделу, и поможет составить школьникам четкое визуальное представление клеток и их органелл.

1 Анализ требований и постановка задачи

1.1 Анализ программных продуктов аналогичного назначения

Для отбора приложений для проведения анализа программных продуктов аналогичного назначения, будет использован Google Play – магазин приложений для операционной системы Android.

Существует множество мобильных приложений по биологии, позволяющих изучить строение клеток. Был проведен анализ наиболее популярных приложений и из них были выбраны следующие приложения: «3D Biology +», «Cell Biology», «Cell World», «Биология.Викторина» и «Inside Cell VR V2». Отбор проводился среди бесплатных приложений.

Приложение «3D Biology +» представлено на площадке «Play Market» от разработчика «Vaibhav Kokare».

В данном приложении доступны следующие действия:

- просмотр модели клетки;
- вращение модели;
- приближение и отдаление клетки.

Информацию о каждой органелле и о самой клетке в целом можно прочитать или прослушать, выбрав интересующий элемент клетки касанием по нему. Аудиоинформация и текстовое описание написаны на английском языке. Также выбранный элемент можно скрыть, чтобы посмотреть, что находится позади него. В приложении также имеется функция «Eagle Eye View». Данная функция позволяет делать модели прозрачными. Модель клетки очень хорошо проработана, все органеллы хорошо различимы и видны.

Основными недостатками приложения «3D Biology +» является то, что в нем не поддерживается русский язык, а также в приложении нельзя проверить свои знания и пройти тестирование.

Интерфейс приложения представлен на рисунке 1.



Рисунок 1 – Интерфейс приложения «3D Biology +»

Приложение «Cell Biology» представлено на площадке «Play Market» от разработчика «Northwest Project Development». Данное приложение предоставляет возможность прохождения тестирования на тему «Клетка». Перед тестированием можно выбрать темы, которые пользователь хотел бы проверить. Также в данном приложении перед прохождением теста пользователь может потренироваться на флэш-карточках с вопросами, которые используются в тесте. Приложение содержит и блок теории, который пользователь может прочитать в любое время.

Минусом данного приложения является то, что в нем не используются 3D-модели клеток и органелл и вся визуальная часть состоит только из картинок, которые не всегда хорошо просматриваются. Также минусом является то, что весь интерфейс приложения на английском языке и соответственно данное приложение не может использоваться российскими школами. Интерфейс приложения «Cell Biology» представлен на рисунке 2.

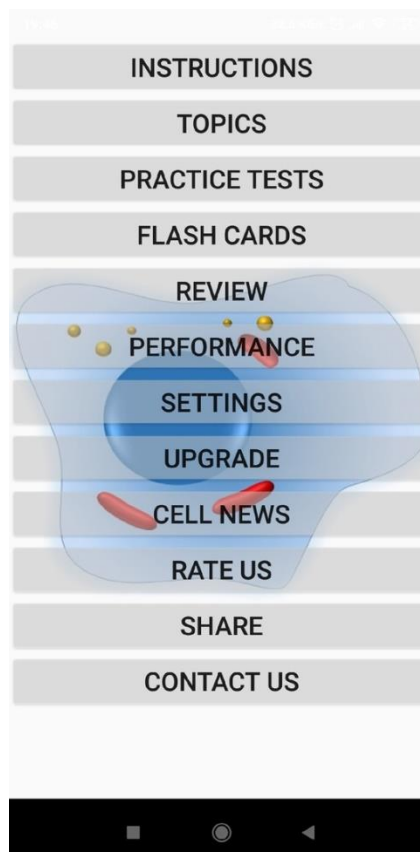


Рисунок 2 – Интерфейс приложения «Cell Biology»

Приложение «Cell World» представлено на площадке «Play Market» от разработчика «V. I. E. W.». Приложение содержит модель клетки с описанием каждой органеллы. Пользователь может прочитать информацию об органелле или же воспользоваться функцией голосового помощника и прослушать ее. Пользователь может слушать информацию об органеллах в порядке, который представлен в приложении или же выбрать органеллу, о которой он хотел бы узнать, кликая на нее.

Приложение содержит функцию масштабирования и вращения, при нажатии на органеллу приложение автоматически приближает ее для лучшего рассмотрения. Модель клетки очень хорошо проработана. Основные минусы приложения: не поддерживается русский язык, нет анимации процессов, происходящих в клетке, а также отсутствует возможность тестирования для проверки знаний.

Интерфейс представлен на рисунке 3.

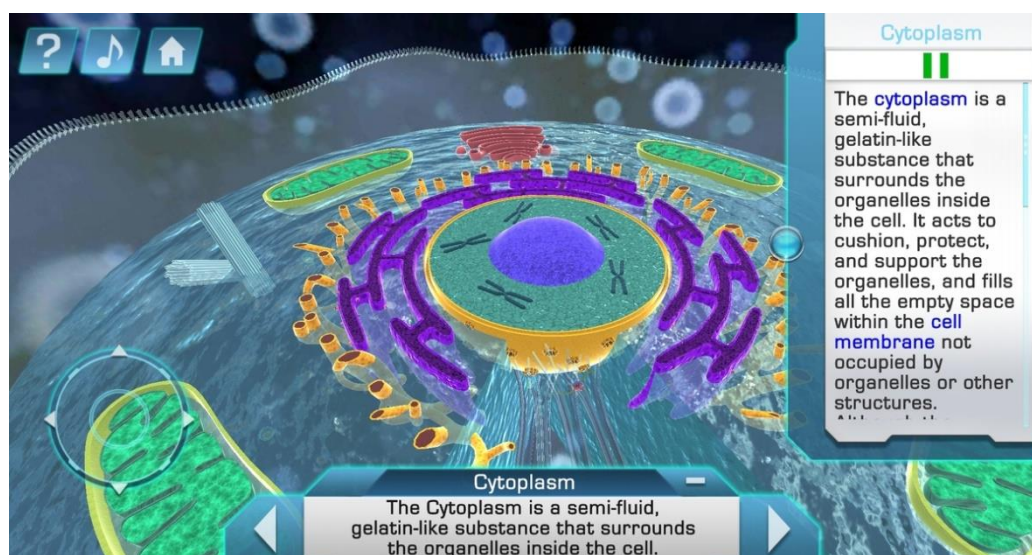


Рисунок 3 – Интерфейс приложения «Cell World»

Приложение «Биология.Викторина» представлено на площадке «Play Market» от разработчика «Yugyd». Данное приложение содержит множество тестов на различные темы по биологии. В их число входит и тема «Клетка». Приложение имеет очень простой и понятный интерфейс на русском языке. Тестирование имеет два вида: марафон и спринт. Марафон предполагает выполнение всех заданий в тесте. Спринт – выполнение наибольшего количества заданий в минуту.

В данном приложении отсутствует теоретический материал по каждому из блоков, что существенно затрудняет изучение темы. Следовательно, пользователь сможет обучаться только по тестированию, анализируя правильность и неправильность собственных ответов.

Минусом данного приложения является отсутствие моделей клеток и теоретической части о функциях органелл. Также в данном приложении затрагиваются все направления в изучении клетки, следовательно, тема строения клетки рассмотрена недостаточно подробно.

Интерфейс приложения «Биология.Викторина» представлен на рисунке 4.

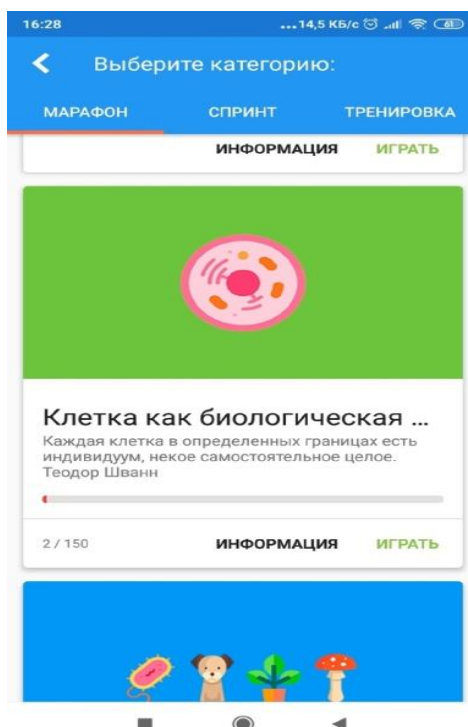


Рисунок 4 – Интерфейс приложения «Биология.Викторина»

Приложение «Inside Cell VR V2» представлено на площадке «Play Market» от разработчика «Scientific Animation Inc». Это VR-приложение, т.е. приложение, которое полностью погружает человека в виртуальную реальность. В данном приложении можно погрузиться внутрь клетки и рассмотреть ее с разных ракурсов. Модели органелл очень хорошо просматриваемы и расположены в различных частях клетки.

В приложении поддерживается голосовой помощник, который рассказывает пользователю об органеллах клетки.

В «Inside Cell VR V2» не поддерживается русский язык, что является огромным минусом, так как данное приложение не подходит для российских школ. В приложении отсутствует проработка анимации биологических процессов, происходящих в клетке, и нет возможности протестировать полученные знания.

Интерфейс приложения представлен на рисунке 5.

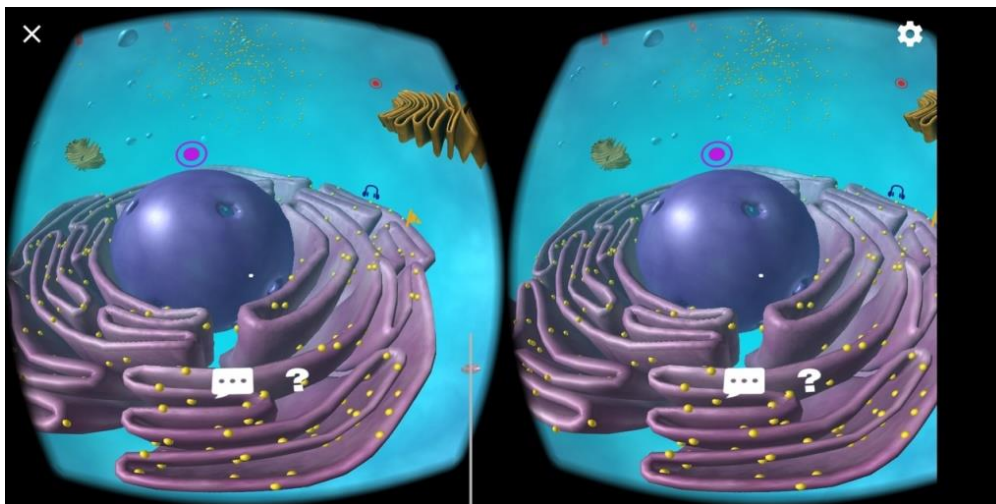


Рисунок 5 – Интерфейс приложения «Inside Cell VR V2»

Рассмотренные программы обладают различными функциональными возможностями и характеристиками. В результате проведенного анализа выделены критерии, напрямую влияющие на эффективность применения приложения в учебном процессе:

- наличие описания клеток и органелл;
- возможность наблюдения реакций и различных процессов;
- наличие 3D-модели клетки;
- дополненная реальность;
- наличие базы проверочных заданий, возможность проведения тестирования;
- удобство приложения в использовании.

Полученные результаты сравнения сведены в единую таблицу (таблица 1).

Таблица 1 – Сравнение программ-аналогов по критериям

Название	3D- модель клетки	Наличие AR/VR	Наблюдение реакций	Наличие базы проверочных заданий	Описание клеток и органелл
3D Biology +	+	-	-	-	+

Продолжение таблицы 1

Название	3D- модель клетки	Наличие AR/VR	Наблюдение реакций	Наличие банка проверочных заданий	Описание клеток и органелл
Cell Biology	-	-	-	+	+
Cell World	+	-	-	-	+
Биология.Викторина	-	-	-	+	-
Inside Cell VR V2	+	+	-	-	+

Просмотрев результаты сравнения, можно сделать вывод, что приложение «Inside Cell VR V2» и «Cell Biology» обладают расширенными функциональными возможностями, более удобным интерфейсом. «Inside Cell VR V2» использует VR-технологии. «Cell Biology» имеет банк заданий и теоретическую составляющую, которые позволяют изучить строение клетки и проверить собственные знания. Но, как показывает анализ программных продуктов аналогичного назначения, ни одно из рассмотренных приложений не объединяет 3D-технологии для изучения строения клетки, возможность наблюдения реакций и функцию тестирования. Также большинство из рассмотренных программных продуктов поддерживают только английский язык.

1.2 Постановка задачи на разработку подсистемы приложения «Живая клетка»

Данная выпускная квалификационная работа направлена на разработку подсистемы приложения дополненной реальности «Живая клетка». Приложение бы позволило изучить модели животной клетки, растительной клетки, а также вируса и бактерии. В приложении также должна быть реализована анимация биологических процессов и реакций, происходящих в

клетке. Приложение должно быть включено в образовательную платформу Experiment, через которую проходила бы авторизация в приложении.

Приложение должно быть разработано с помощью технологий дополненной реальности и работать на устройствах с операционной системой Android выше версии 7.0 и поддерживающих AR-технологии.

Приложение должно иметь понятный и удобный для пользователя интерфейс. Так как приложение является образовательным, то модели клеток должны быть хорошо проработаны, органеллы должны быть четко различимы, должно присутствовать название каждой органеллы и ее описание. При входе в приложение пользователю должно быть выведено сообщение о том, что в данном приложении используется технология дополненной реальности, а также уведомления о том, что приложению требуется разрешение на доступ к камере. Если пользователь не зарегистрирован в приложении Experiment Starter и не вошел в него, то должно быть показано уведомление о том, что для начала работы в приложении «Живая клетка» нужно зарегистрироваться и войти в систему.

Также приложение должно предоставлять пользователю возможность проверки своих знаний путем тестирования, а также прохождения индивидуальных заданий по изучению клетки. Но в данной выпускной квалификационной работе будет рассмотрена разработка клиентской части приложения «Живая клетка».

Таким образом, можно выделить основные подзадачи на разработку подсистемы приложения дополненной реальности «Живая клетка»:

- изучение учебной литературы по биологии для создания моделей клеток;
- изучение дополнительной литературы для разработки приложения;
- создание моделей клеток;
- проектирование приложения;
- создание структуры приложения;
- реализация клиентской части приложения дополненной реальности.

После того, как данные задачи будут выполнены, будет создана и оформлена документация по приложению «Живая клетка».

1.3 Анализ функциональных требований

Подсистема приложения дополненной реальности «Живая клетка» предназначено для изучения строения животной и растительной клеток, вируса и бактерии, их органелл и взаимодействия органелл друг с другом. Для разрабатываемого приложения можно выделить следующие функциональные требования:

- в приложении должна использоваться технология дополненной реальности;
- приложение должно определять положение устройства в пространстве;
- приложение должно позволять пользователю добавлять 3D-объекты на AR-сцену и масштабировать их.

Существует два вида пользователей, которые будут использовать данное приложение: ученик и учитель.

Для отображения функций данных действующих лиц была построена диаграмма вариантов использования, которая приведена на рисунке 6.



Рисунок 6 – Диаграмма вариантов использования приложения «Живая клетка»

В данной выпускной квалификационной работе будет рассмотрена разработка клиентской части приложения «Живая клетка». Диаграмма вариантов использования для разрабатываемой подсистемы приложения представлена на рисунке 7.

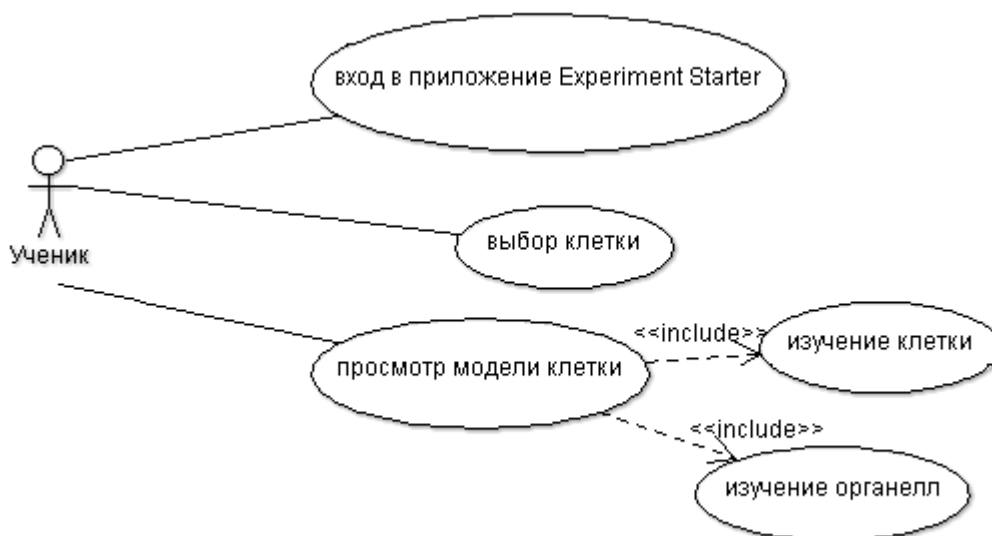


Рисунок 7 – Диаграмма вариантов использования для подсистемы приложения «Живая клетка»

Краткое описание действующего лица, изображенного на диаграмме вариантов использования подсистемы приложения, представлено в таблице 2.

Таблица 2 – Описание действующего лица

Название	Описание
Ученик	Зарегистрированный пользователь приложения. Использует приложения для изучения раздела биологии «Цитология». Имеет доступ к базовым функциям приложения «Живая клетка».

Для ученика можно выделить следующие варианты использования:

- вход в приложение Experiment Starter;
- выбор клетки;
- просмотр модели клетки;

- изучение клетки;
- изучение органелл.

Краткое описание вариантов использования описано в таблице 3.

Таблица 3 – Краткое описание вариантов использования

Действующее лицо	Цель	Краткое описание
Ученик	Вход в приложение Experiment Starter	Ученик перед тем, чтобы войти в приложение «Живая клетка», должен войти в аккаунт в приложении Experiment Starter.
Ученик	Выбор клетки	Ученик, войдя в приложение, из предложенного меню выбирает клетку, которую он бы хотел изучить.
Ученик	Просмотр модели клетки	Ученик рассматривает выбранную клетку.
Ученик	Изучение органелл	Ученик изучает органеллы клетки.
Ученик	Изучение клетки	Ученик изучает строение выбранной клетки.

Описание прецедента «Вход в приложение Experiment Starter» приведено в таблице 4.

Таблица 4 – Описание прецедента «Вход в приложение Experiment Starter»

Название пункта	Описание
Наименование	Вход в приложение Experiment Starter
Краткое описание	Ученик перед тем, чтобы войти в приложение «Живая клетка», должен войти в аккаунт в приложении Experiment Starter.
Главные актеры	Ученик
Второстепенные актеры	Нет
Предусловие	Ученик решил войти в приложение «Живая клетка»
Основной поток	1. Ученик авторизуется в приложении
Постусловие	Ученик авторизовался в приложении Experiment Starter
Альтернативные потоки	<ul style="list-style-type: none"> - Ученик не зарегистрирован в приложении - Ученик не скачал приложение - Ученик скачивает и проходит авторизацию в приложении

Описание варианта использования «Выбор клетки» представлено в таблице 5.

Таблица 5 – Описание прецедента «Выбор клетки»

Название пункта	Описание
Наименование	Выбор клетки
Краткое описание	Ученик, войдя в приложение, из предложенного меню выбирает клетку, которую он бы хотел изучить.
Главные актеры	Ученик
Второстепенные актеры	Нет
Предусловие	Ученик вошел в приложение Experiment Starter
Основной поток	1. Ученик вошел в приложение 2. Ученик выбирает клетку для просмотра
Постусловие	Ученик выбрал клетку для просмотра
Альтернативные потоки	-

Описание прецедента «Просмотр модели клетки» представлено в таблице 6.

Таблица 6 – Описание прецедента «Просмотр модели клетки»

Название пункта	Описание
Наименование	Просмотр модели клетки
Краткое описание	Ученик рассматривает выбранную клетку.
Главные актеры	Ученик
Второстепенные актеры	Нет
Предусловие	Ученик выбрал клетку для просмотра
Основной поток	1. Ученик рассматривает клетку
Постусловие	Ученик рассмотрел клетку
Альтернативные потоки	-

Описание варианта использования «Изучение клетки» приведено в таблице 7.

Таблица 7 – Описание варианта использования «Изучение клетки»

Название пункта	Описание
Наименование	Изучение клетки
Краткое описание	Ученик изучает строение выбранной клетки.
Главные актеры	Ученик
Второстепенные актеры	Нет
Предусловие	Ученик выбрал клетку для просмотра
Основной поток	1. Ученик изучает клетку и ее строение
Постусловие	Ученик изучил клетку и ее строение
Альтернативные потоки	-

Описание варианта использования «Изучение органелл» представлено в таблице 8.

Таблица 8 – Описание прецедента «Изучение органелл»

Название пункта	Описание
Наименование	Изучение органелл
Краткое описание	Ученик изучает органеллы клетки.
Главные актеры	Ученик
Второстепенные актеры	Нет
Предусловие	Ученик изучил клетку
Основной поток	1. Ученик изучает органеллы и их строение
Постусловие	Ученик изучил органеллы и их строение
Альтернативные потоки	-

Таким образом, функциональные требования к приложению «Живая клетка» были объединены и изображены графически с помощью диаграммы вариантов использования.

1.4 Выбор средств разработки

Для разработки подсистемы приложения дополненной реальности «Живая клетка» был выбран язык программирования Java, а в качестве среды разработки – среда Android Studio.

Выбор языка программирования Java обусловлен тем, что Java – объектно-ориентированный язык и позволяет использовать инструменты парадигмы объектно-ориентированного программирования. Также, данный язык был выбран потому, что Java является самым популярным языком, используемым для разработки мобильных приложений на операционную систему Android.

Помимо этого, Java обладает следующими преимуществами:

- при компиляции программы создается байт-код, который преобразуется в команды, характерные для определенной операционной системы;
- позволяет не прописывать уничтожение объектов, которые более не используются в программе, что позволяет сократить время на разработку [2].

Android Studio – среда разработки мобильных приложений для мобильных устройств с операционной системой Android. Выбор данного ПО обусловлен тем, что приложение дополненной реальности «Живая клетка» будет поддерживаться на Android-устройствах. Важными плюсами Android Studio также являются: наличие эмулятора, который может использоваться для тестирования приложения на различных устройствах, рефакторинг кода, а также наличие средств для разработки приложений с дополненной реальностью.

Для проектирования приложения были использованы CASE-средство ERwin Data Modeler 7.3 и ArgoUML. Для проектирования структур данных была использована нотация IDEF1X, а для проектирования программных средств использовался унифицированный язык моделирования UML.

Для планирования проекта и расчета бюджета был использован Microsoft Project, который позволил внести используемые ресурсы, рассчитать их использование и организовать продуктивную и своевременную работу над проектом.

Для создания моделей клеток было использовано программное обеспечение Blender, которое позволяет работать с объемными графическими объектами и имеет большие функциональные возможности для создания реалистичных объектов.

2 Проектирование подсистемы приложения дополненной реальности «Живая клетка»

2.1 Функциональное моделирование приложения «Живая клетка»

Функциональное моделирование – это моделирование функций, реализуемых в разрабатываемой системе. Функции разрабатываемой системы изображаются в виде графических блоков. Функциональное моделирование осуществляется с помощью методологии SADT (Structured Analysis and Design Technique). На основе SADT была разработана методология IDEF0. Методология IDEF0 – позволяет структурно описать бизнес-процессы разрабатываемой программы, т.е. деятельность программы, которая направлена на определенный результат. Диаграмма, построенная по методологии IDEF0, имеет иерархический характер.

Далее будет представлена диаграмма, отображающая бизнес-процессы приложения дополненной реальности «Живая клетка». Функции приложения рассмотрены со стороны пользователя данного приложения. Контекстная диаграмма изображена на рисунке 8.

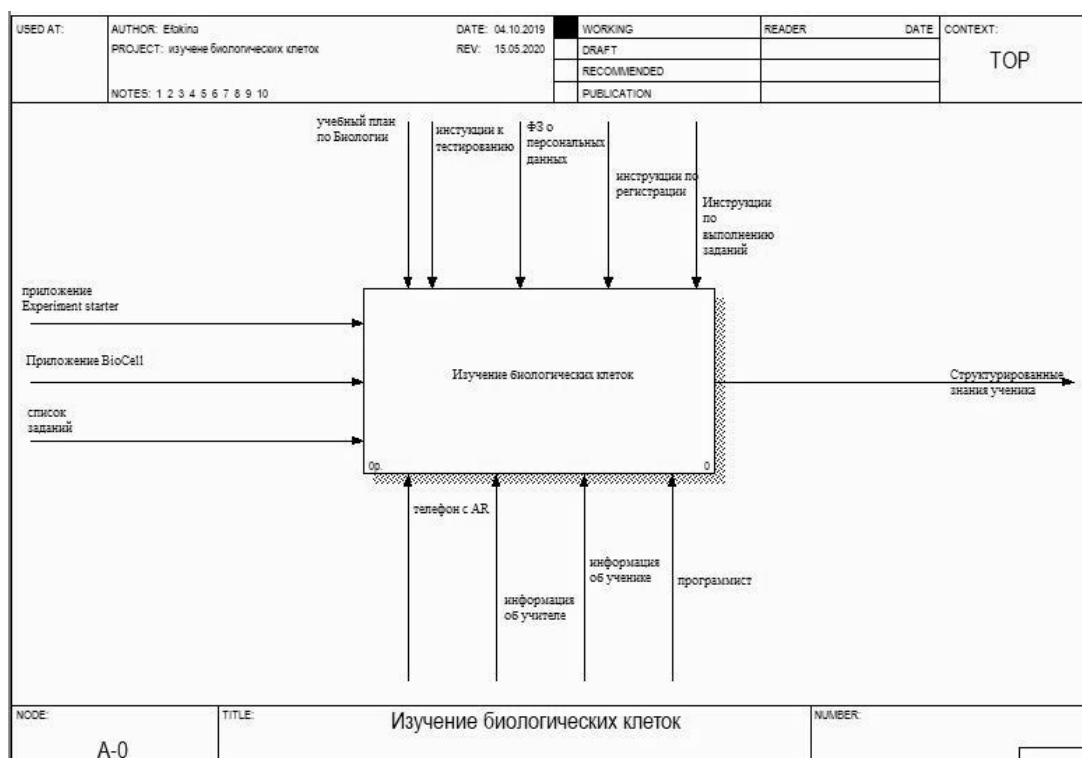


Рисунок 8 – Контекстная диаграмма в нотации IDEF0

С помощью функционального блока на контекстной диаграмме отображено основное назначение разрабатываемого приложения. Так как методология IDEF0 заключается в декомпозиции функциональных блоков, то на следующем рисунке (рисунок 9) будет изображена дочерняя диаграмма следующего уровня.

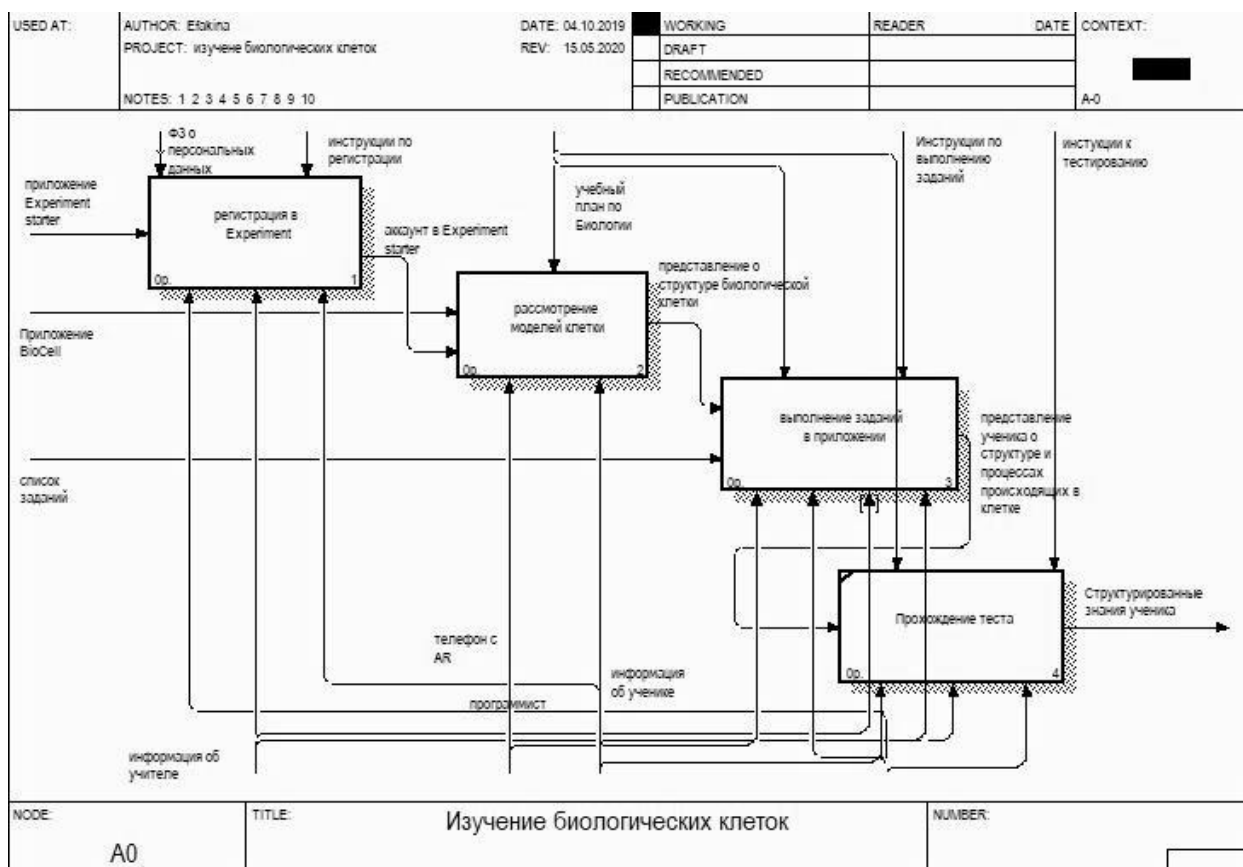


Рисунок 9 – Диаграмма декомпозиции изучения биологических клеток

Пользователь приложения «Живая клетка» (ученик) регистрируется в приложении Experiment Starter, после чего он может войти в приложение и рассмотреть различные модели клеток. Помимо этого, пользователь проходит задания, связанные с темой строения биологических клеток, а также биологических процессов, проходящих в них. Перед тем, как ученик сможет пройти задания по выбранной теме, должен быть создан список заданий. Список заданий создает учитель на сайте Experiment School. Пройдя задания, ученик должен пройти тест, проверяющий полученные знания.

Функциональные блоки на диаграмме декомпозиции изучения биологических клеток также будут декомпозированы. Декомпозиция блока «Регистрация в Experiment» приведена на рисунке 10.

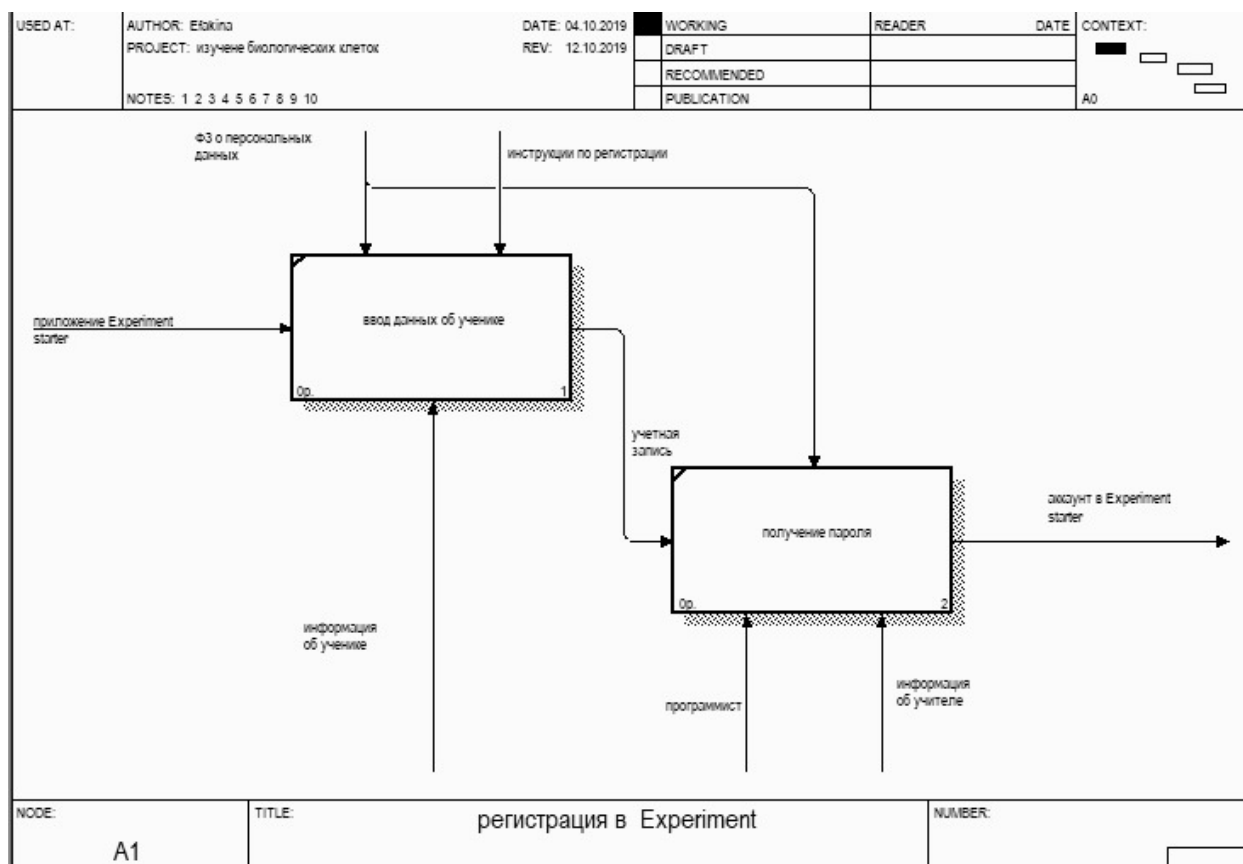


Рисунок 10 – Диаграмма декомпозиции «Регистрация в Experiment»

Для регистрации в приложении Experiment Starter,ученику нужно ввести в специальное поле информацию о себе, после чего программист выдаст пароль для входа в приложение.

Следующие функциональные блоки будут декомпозированы и описаны с помощью методологий DFD и IDEF3.

IDEF3 – это способ описания процессов, который позволяет эксперту в предметной области представить процессы в виде упорядоченной последовательности событий. Методология IDEF3 определяет сценарий процессов, проходящих в разрабатываемой системе.

Декомпозиция блока «Рассмотрение моделей клетки» с помощью методологии IDEF3 приведена на рисунке 11.

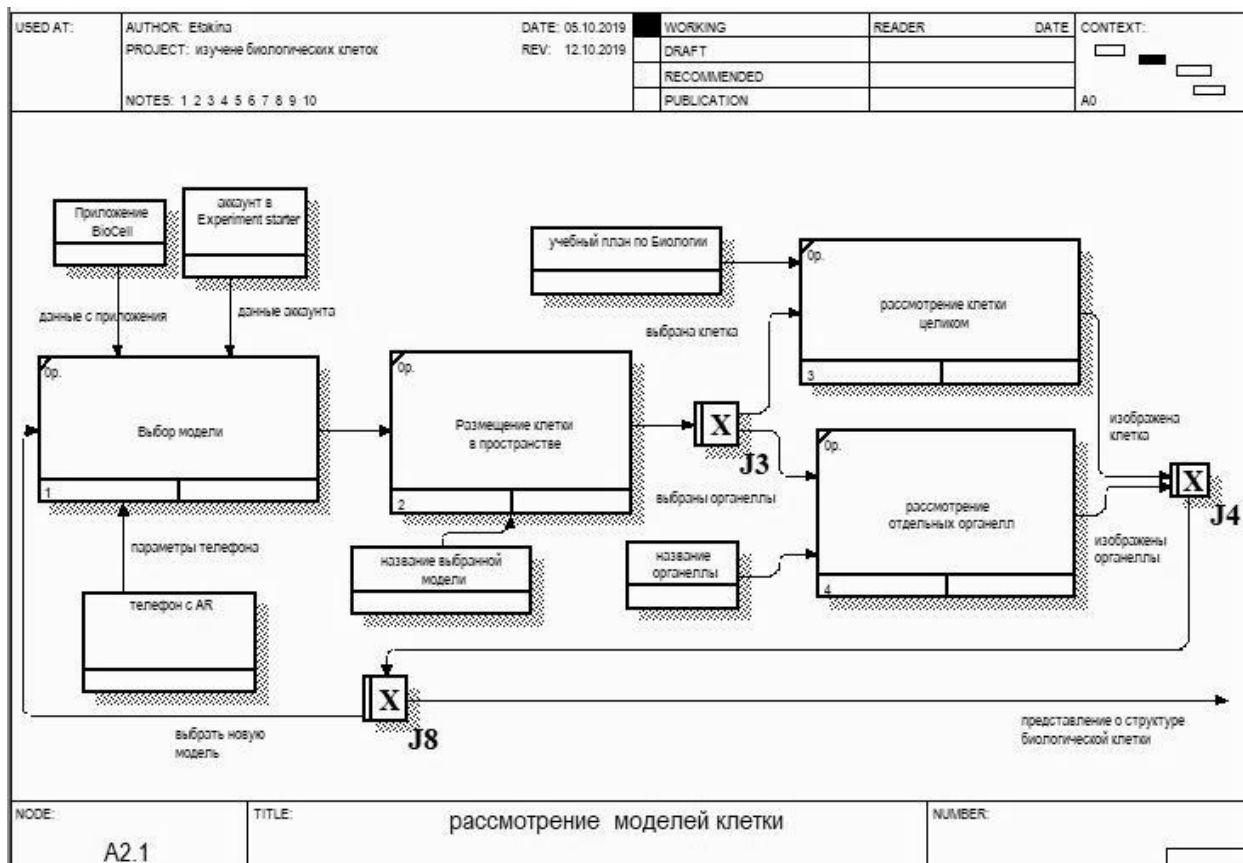


Рисунок 11 – Диаграмма декомпозиции «Рассмотрение моделей клетки»

Процесс рассмотрения модели клетки можно разделить на несколько подпроцессов: выбор модели, размещение клетки в пространстве, рассмотрение клетки целиком и рассмотрение отдельных оргanelл.

Процесс выполнения заданий в приложении будет декомпозирован с помощью методологии DFD (рисунок 12).

Методология DFD – это методология, с помощью которой описываются диаграммы потоков данных.

Диаграммы потоков данных показывают, как входные данные преобразуются в выходные, а также выявляют отношения между процессами. DFD представляет моделируемую систему как сеть связанных работ. Поток данных определяет информационный поток, который идет от отправителя к получателю. Диаграмма потоков данных не показывает последовательность работ.

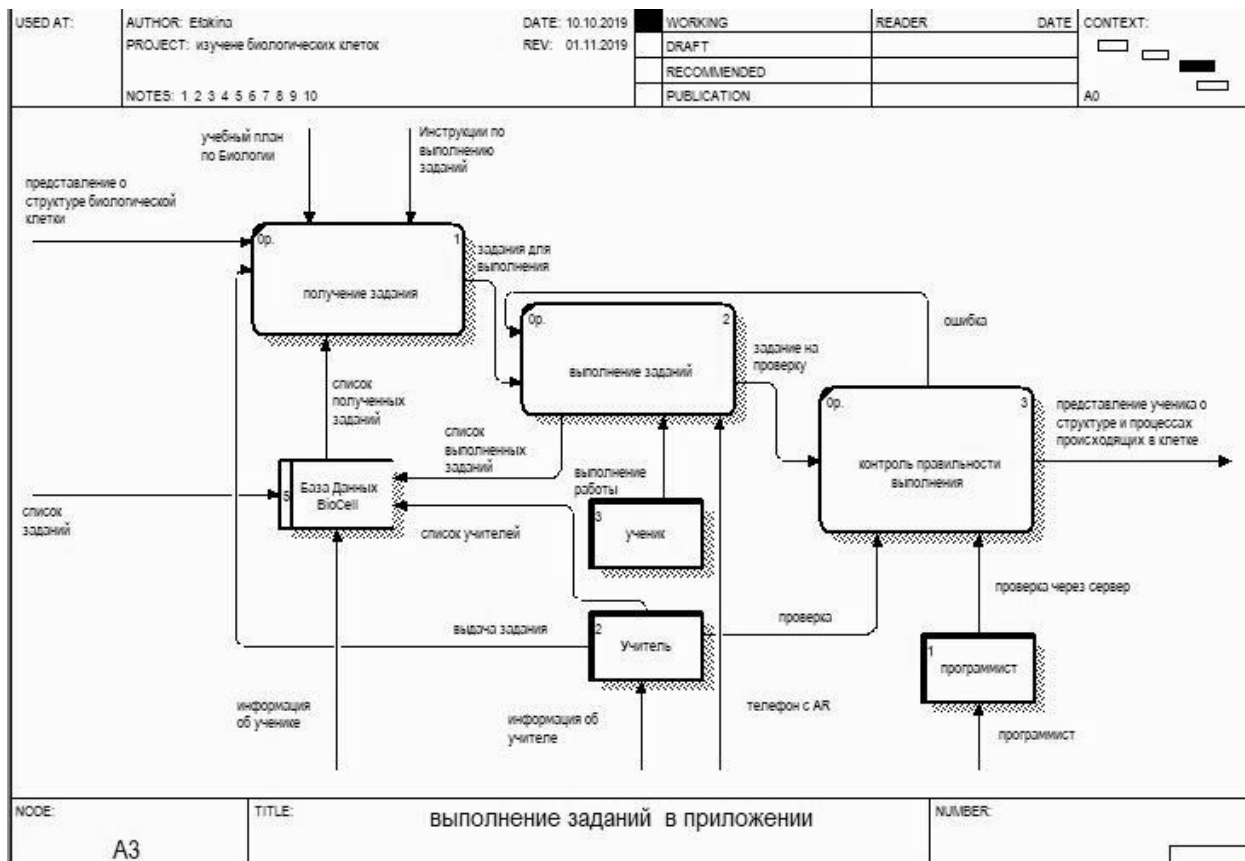


Рисунок 12 – Диаграмма декомпозиции «Выполнение заданий в приложении» в нотации DFD

Учитель создает список заданий для конкретного ученика или класса. Ученик получает эти задания и переходит к их выполнению. После этого, проверяется правильность выполненных учеником заданий. Правильность выполнения заданий проверяется в приложении автоматически, учитель может посмотреть результаты прохождения тестов на сайте Experiment School.

Для функционального моделирования и представления бизнес-процессов разрабатываемого приложения дополненной реальности было использовано CASE-средство AllFusion Process Modeler (BPwin).

2.2 Проектирование структур данных

Структура данных – это эффективный способ хранения и обработки однотипных или связанных данных на компьютере. Проектирование структур данных является необходимым этапом проектирования

приложения. Проектирование структур данных – это процесс создания схемы базы данных и определения необходимых ограничений целостности [3].

Проектирование структуры данных осуществлялось с помощью Case-средства ERwin Data Modeler 7.3 и его методологии IDEF1X. IDEF1X является методом для разработки реляционных баз данных и имеет синтаксис, специально разработанный для построения концептуальных схем. Концептуальной схемой называют универсальное представление структуры данных в рамках предприятия, независимое от конечной реализации базы данных и аппаратной платформы.

Для создания структуры данных в стандарте IDEF1X были рассмотрены два уровня представления модели данных – логический и физический.

2.2.1 Логическая модель базы данных

Логическая модель представляет собой модель базы данных, которая не привязана к конкретной СУБД. В ней выделяют основные объекты БД и определяют связи между этими объектами.

Логическая модель базы данных приложения дополненной реальности «Живая клетка» представлена на рисунке 13.

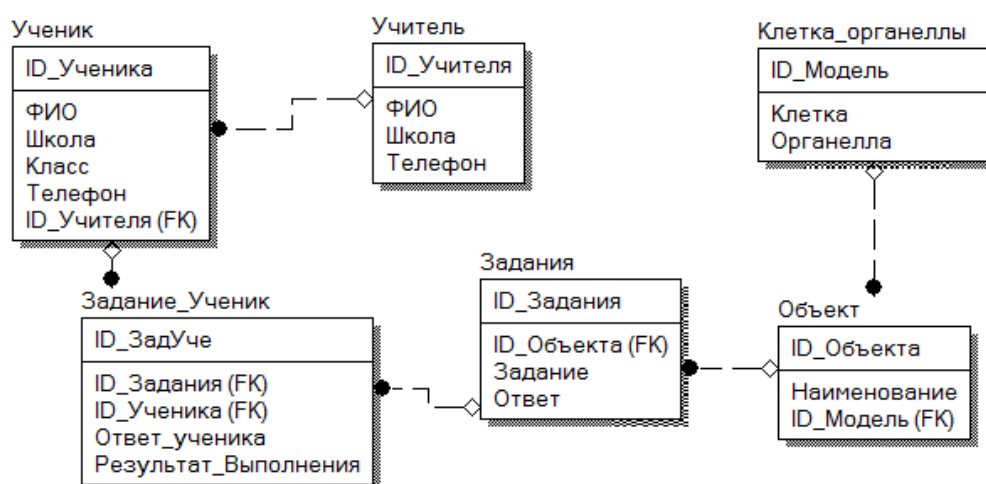


Рисунок 13 – Логическая модель базы данных приложения «Живая клетка»

После того, как были созданы сущности, атрибуты и логические связи между сущностями, следует приступить к проектированию физической модели базы данных.

2.2.2 Физическая модель базы данных

При переходе от логической к физической модели реляционной БД нужно будет работать с объектами и ограничениями, поддерживаемыми в выбранной СУБД. Для построения диаграмм физической модели базы данных в среде ERwin также используется нотация IDEF1X.

Физическая модель базы данных приложения дополненной реальности «Живая клетка» представлено на рисунке 14.

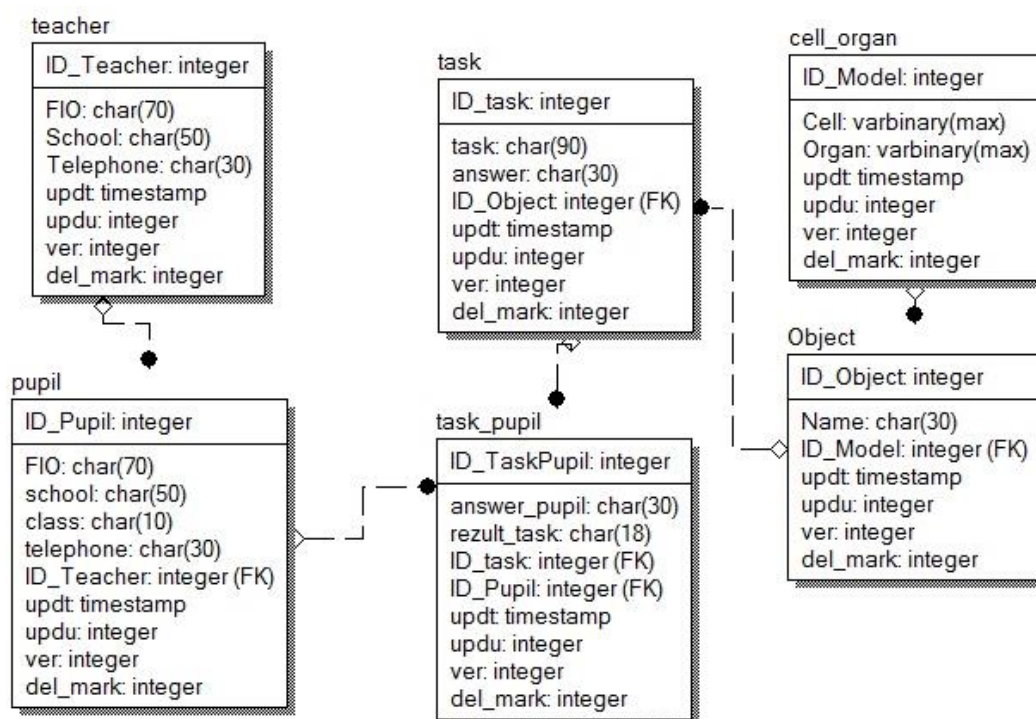


Рисунок 14 – Физическая модель базы данных приложения «Живая клетка»

В физической модели были добавлены типы данных атрибутов, а также были добавлены такие атрибуты, как:

- updt – атрибут, который нужен для определения даты и времени обновления базы, он используется в ситуациях, когда нужно отменить произведенные изменения;

- updu – атрибут, сохраняющий ID пользователя, который обновлял базу последним;
- ver – версия обновления;
- del_mark – пометка на удаление.

2.3 Проектирование программных средств разрабатываемого приложения

2.3.1 Диаграмма классов приложения «Живая клетка»

Диаграмма классов (class diagram) служит для представления классов системы, их атрибутов, методов, а также взаимосвязей между классами. На данной диаграмме не указывается время функционирования разрабатываемой системы.

На рисунке 15 изображена диаграмма классов приложения дополненной реальности «Живая клетка».

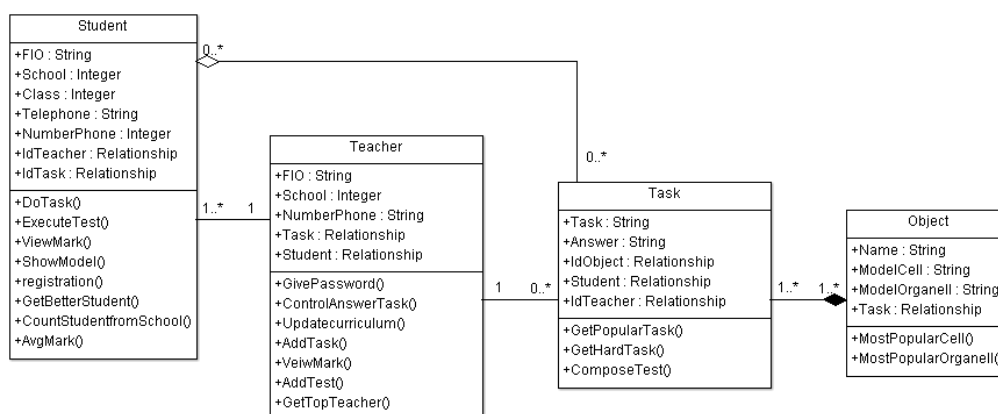


Рисунок 15 – Диаграмма классов приложения дополненной реальности «Живая клетка»

Описание атрибутов и методов классов приложения дополненной реальности «Живая клетка» представлено в таблице 9.

Таблица 9 – Описание атрибутов и методов класса Object

Название	Описание
Имя класса	Object

Продолжение таблицы 9

Название	Описание
Назначение	Описывает список объектов
Атрибуты	Name: String(70) – Название объекта ModelCell: String(40) – модель клетки ModelOrganell: String(40) – модель органеллы Task: Relationship – связь с классом задание
Методы	MostPopularCell() – получить самую популярную клетку MostPopularOrganell() – получить самую популярную органеллу
Связь	Task - связь композиция. Задание имеет привязку к объекту, которое имеет жесткую зависимость времени существования экземпляра класса Объект. Если будет удален объект, то и удалится задание.

Описание атрибутов и методов класса Student приведено в таблице 10.

Таблица 10 – Описание атрибутов и методов класса Student

Название	Описание
Имя класса	Student
Назначение	Описывает ученика, обучающегося в приложении
Атрибуты	FIO: String(70) – фамилия имя отчество студента. School: String(35) – школа ученика Class: String(10) – класс ученика. Telephone: String(30) – модель телефона ученика NumberPhone: String(35) – номер телефона ученика IdTeacher : Relationship – Связь с классом учитель IdTask: Relationship – Связь с классом задание
Методы	DoTask() – выполнение заданий ExecuteTest() – выполнение теста ViewMark() – просмотр оценок ShowModel() – просмотр моделей Registration() - регистрация GetBetterStudent() – вывод лучших учеников CountStudentfromSchool() – подсчет учеников из одной школы AvgMark() – средний бал

Продолжение таблицы 10

Название	Описание
Связь	Teacher – связь Ассоциация. Ассоциация показывает, что объекты класса Student связаны с объектами класса Teacher таким образом, что можно перемещаться от объектов одного класса к другому.
	Task – связь Агрегация. Класс Student включает в себя несколько заданий. При добавлении задания, происходит выдача его ученику и привязка к нему

Описание атрибутов и методов класса Teacher приведено в таблице 11.

Таблица 11 – Описание атрибутов и методов класса Teacher

Название	Описание
Имя класса	Teacher
Назначение	Описывает учителя.
Атрибуты	<p>FIO: String (70) – фамилия имя отчество учителя.</p> <p>School: String(35) – школа учителя</p> <p>NumberPhone: String(35) – номер телефона учителя</p> <p>Student : Relationship – Связь с классом студент</p> <p>Task: Relationship – Связь с классом задание</p>
Методы	<p>GivePassword() – выдача пароля</p> <p>ControlAnswerTask() – проверка ответа ученика на задание</p> <p>UpdateCurriculum() – обновление учебного плана</p> <p>AddTask() – добавление задания</p> <p>AddTest() – добавление теста</p> <p>ViewMark() – просмотреть оценки</p> <p>GetTopTeacher() – список лучших учителей</p>
Связь	<p>Teacher связаны с объектами класса Student таким образом, что можно перемещаться от объектов одного класса к другому.</p> <p>Task - связь Ассоциация. Ассоциация показывает, что объекты класса Teacher связаны с объектами класса Task таким образом, что можно перемещаться от объектов одного класса к другому.</p>

Описание атрибутов и методов класса Task приведено в таблице 12.

Таблица 12 – Описание атрибутов и методов класса Task

Название	Описание
Имя класса	Task
Назначение	Описывает список заданий.
Атрибуты	Task: String(70) – Описание задания Answer: String(50) – ответ на задание IdObject : Relationship – Связь с классом объект Student : Relationship – Связь с классом студент IdTeacher : Relationship – Связь с классом учитель
Методы	GetPopularTask() – Вывод списка популярных заданий GetHardTask() – Вывод списка сложнееших заданий ComposeTask() – Составление теста
Связь	Student - связь Агрегация. Класс Student включает в себя несколько заданий. При добавлении задания, происходит выдача его ученику и привязка к нему Object – связь композиция. Задание имеет привязку к объекту , которое имеет жесткую зависимость времени существования экземпляра класса Объект. Если будет удален объект, то и удалится задание. Teacher - связь Ассоциация. Ассоциация показывает, что объекты класса Task связаны с объектами класса Teacher таким образом, что можно перемещаться от объектов одного класса к другому.

Атрибуты класса определяют структуру данных, характерных для каждого объекта класса. Каждый атрибут имеет имя и тип, определяющий, какие данные какого типа он содержит.

2.3.2 Диаграмма состояний для класса Task приложения «Живая клетка»

Диаграмма состояний показывает, как объект класса переходит из одного состояния в другое [4]. Диаграмма состояний удобна в моделировании жизненного цикла объекта.

От других диаграмм диаграмма состояний отличается тем, что описывает процесс изменения состояний только одного экземпляра определенного класса.

Диаграмма состояний класса «Task» представлена на рисунке 16.

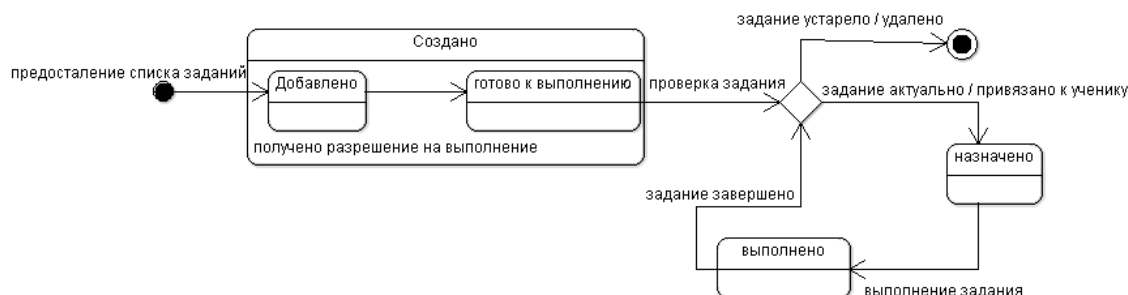


Рисунок 16 – Диаграмма состояний класса «Task»

Диаграмма описывает состояния, в которых находится объект класса Task на протяжении его использования. Состояние «Создано» обозначает, что задание уже создано и включает в себя два других состояния: «Добавлено» и «Готово к выполнению», первое из которых обозначает то, что задание добавлено в приложение, а второе, что задание готово к использованию. После этого идет проверка задания на его актуальность. Если задание устарело, то оно удаляется. Если же оно еще актуально, оно распределяется ученикам. Состояние «Назначено» обозначает, что задание привязано к ученику. Состояние «Выполнено» обозначает, что выполнение задания завершено.

После того, как учитель предоставит список заданий, программист добавляет их в приложение и дает разрешения на их выполнение.

2.3.3 Диаграмма деятельности процесса разработки анимации процессов, происходящих в клетке

Диаграмма деятельности (англ. activity diagram) – диаграмма, описывающая последовательность действий определенного процесса [4]. Диаграмма деятельности используется для подробного описания вариантов использования. Диаграмма деятельности позволяет показать жизненный цикл состояния деятельности объектов.

На рисунке 17 изображена диаграмма деятельности, описывающая процесс разработки анимации процессов, происходящих в клетке.

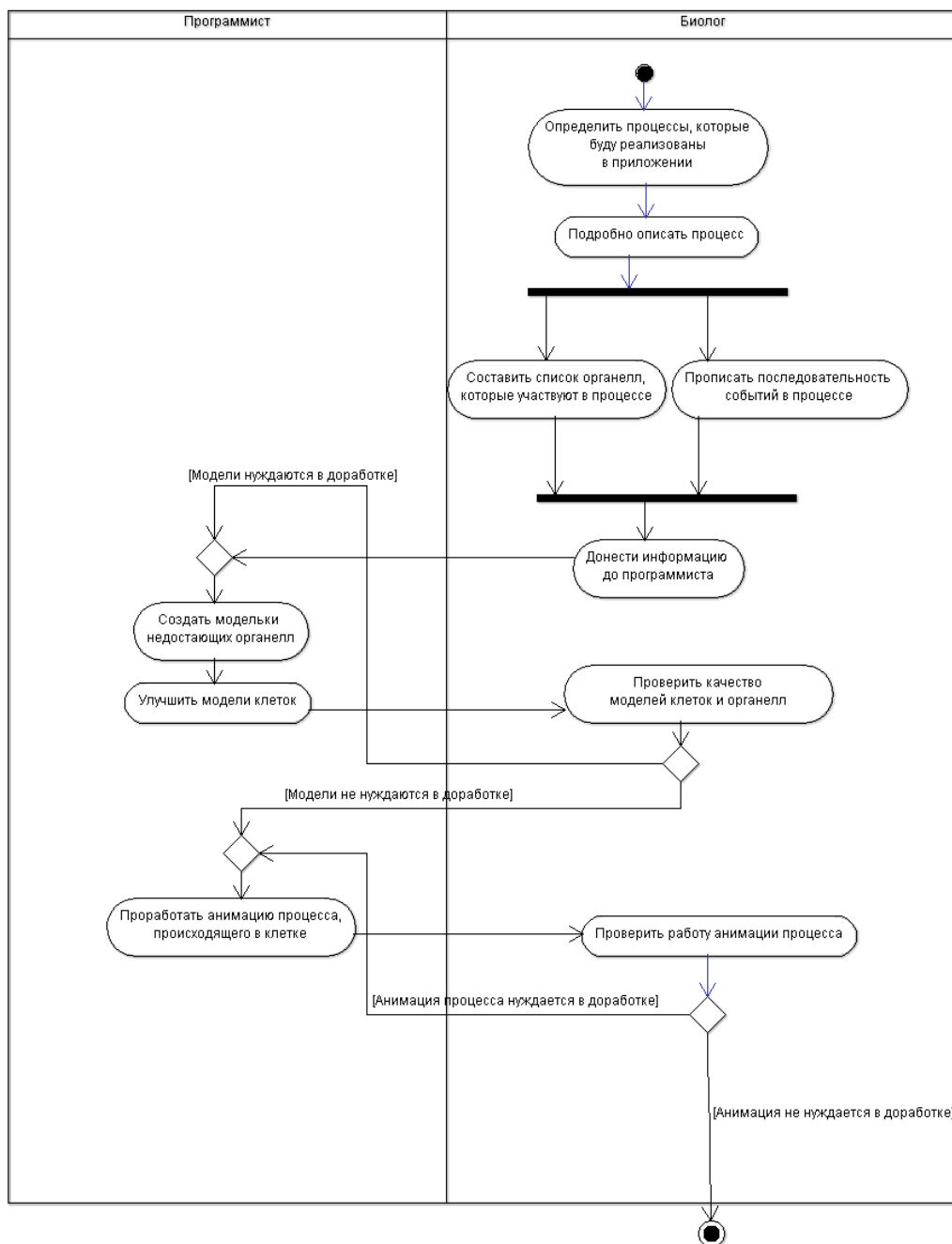


Рисунок 17 – Диаграмма деятельности процесса разработки анимации процессов, происходящих в клетке

Диаграмма деятельности состоит из двух дорожек, которые представляют собой биолога и программиста. Биолог занимается подготовкой информации по процессам, происходящим в клетке.

Сначала биолог определяет процессы, которые будут реализованы в приложении, затем подробно описывает каждый процесс. После этого задачей биолога является составление списка органелл, участвующих в процессе, и описание последовательности событий в процессе.

Программист начинает свою работу после того, как биолог подготовит всю информацию о процессах. Сначала программист создает модели недостающих органелл (модели которых еще не были созданы), затем программист улучшает модели клеток и отдает модели на проверку биологу. Если биолог утверждает модели, то программист начинает разработку анимации процессов. Если модели нуждаются в доработке, то программист занимается доработкой моделей.

После того, как программист разработает анимацию процессов, биолог проверяет результат. Если нужны доработки, то программист улучшает анимацию. Если доработки не требуются, то разработка анимации процессов завершается.

2.3.4 Диаграмма последовательности процесса разработки анимации процессов, происходящих в клетке

Диаграмма последовательности описывает сценарий какого-либо процесса, описанного на диаграмме вариантов использования, показывает жизненный цикл объектов сценария и их взаимодействие.

В отличие от диаграммы деятельности данная диаграмма позволяет отобразить сообщения, имитирующие вызов определенных функций. Диаграмма последовательности является одной из диаграмм взаимодействия.

Диаграмма последовательности также может использоваться не только при проектировании системы, но и в других ситуациях, которые можно показать в виде сценария, например сетевое и межпроцессорное взаимодействие.

Как было замечено ранее, диаграмма последовательности описывает объекты и обмен сообщениями между ними. Объектами обычно являются классы системы, которые обладают определенным поведением.

Диаграмма последовательности, описывающая процесс разработки анимации процессов, происходящих в клетке, приведена на рисунке 18.

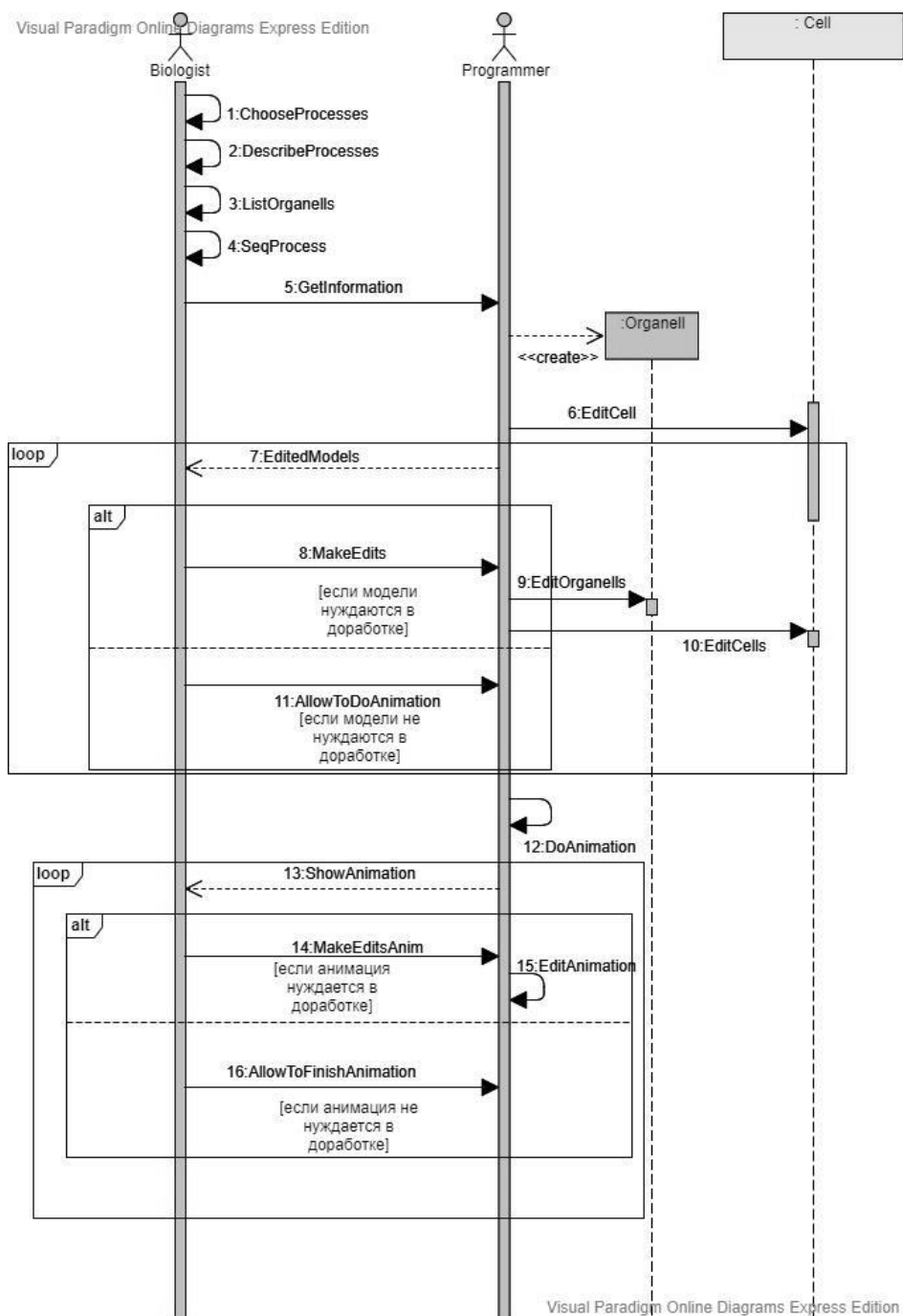


Рисунок 18 – Диаграмма последовательности процесса разработки анимации процессов, происходящих в клетке

Диаграмма включает в себя четыре объекта: программист, биолог, клетка и органелла, которые взаимодействуют в рамках сценария и обмениваются сообщениями.

Сначала биолог определяет процессы, которые будут анимированы, подробно описывает их, составляет список органелл, участвующих в процессах, прописывает последовательность процесса и затем передает эту информацию программисту. Программист, в свою очередь, создает органеллы, которые еще не созданы, но участвуют в процессе, и редактирует модели клеток, которые были созданы до этапа разработки анимации. После этого, программист показывает отредактированные модели биологу, биолог может внести какие-либо правки, либо одобрить данные модели. Если биолог вносит правки, то программист исправляет недостатки. Если модели одобрены, биолог дает разрешение на разработку анимации. Программист разрабатывает анимацию, а затем показывает результат биологу. Биолог вносит правки или же принимает такую анимацию. Если анимация не нуждается в доработке, то процесс разработки анимации будет завершен.

2.3.5 Диаграмма кооперации для процесса создания моделей клеток

Диаграмма кооперации является альтернативным диаграмме последовательностей способом представления взаимодействия объектов. Она показывает потоки данных между объектами классов, что позволяет уточнить отношения между ними.

На диаграмме изображаются участвующие во взаимодействии объекты (в виде прямоугольников, содержащих имя объекта, имя его класса и значения атрибутов, если они имеются), а также указываются ассоциации между этими объектами. Дополнительно могут быть изображены динамические связи — потоки сообщений. Обозначения объектов на диаграмме кооперации схожи с обозначением объектов на диаграмме последовательности.

Диаграмма кооперации также относится к диаграммам взаимодействия.

Процесс создания моделей клеток представлен на диаграмме кооперации (рисунок 19).

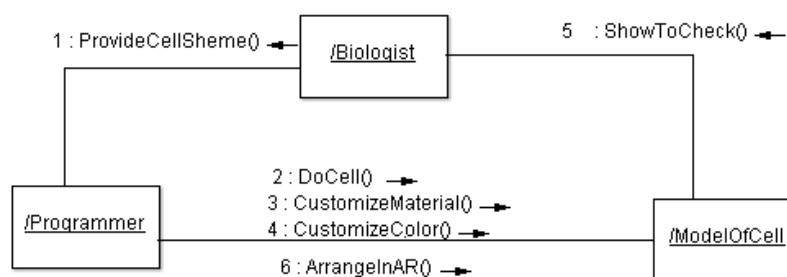


Рисунок 19 – Диаграмма кооперации процесса «Создание моделей клеток»

Процесс создания модели клетки включает в себя три объекта: биолога, программиста и модель клетки, которые обмениваются сообщениями.

После того, как биолог предоставит программисту схему клетки, программист начинает моделировать модель клетки, после этого он настраивает материалы и цвета для модели. Затем модель клетки показывается биологу. После этого, программист располагает полученную клетку в дополненной реальности.

2.4 Проектирование интерфейса приложения «Живая клетка»

Интерфейс пользователя – совокупность различных средств взаимодействия пользователя с приложением [5]. Пользовательский интерфейс должен быть понятным и удобным для пользователя, взаимодействия пользователя с приложением должно быть интуитивным.

Существует несколько критериев проектирования интерфейса мобильного приложения:

- структурированность и взаимосвязь между элементами интерфейса;
- объединение в группы логически связанных элементов;
- выравнивание элементов интерфейса;
- единый стиль интерфейса в целом [6].

Интерфейс клиентской части приложения дополненной реальности «Живая клетка» должен содержать четыре виджета-кнопки, которые отвечают за переход к различным моделям: животной клетке, растительной клетке, бактерии и вируса.

Интерфейс клиентской части приложения дополненной реальности «Живая клетка» представлен на рисунке 20.

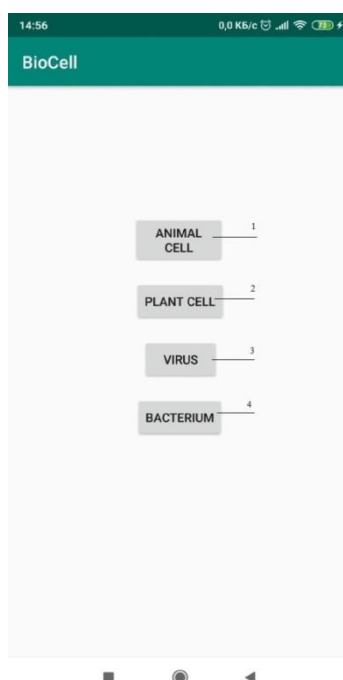


Рисунок 20 – Интерфейс клиентской части приложения дополненной реальности «Живая клетка»

Кнопки-виджеты отвечают за переход на AR-сцену и загрузку определенных моделей клеток:

- 1 – кнопка загрузки модели животной клетки;
- 2 – кнопка загрузки модели растительной клетки;
- 3 – кнопка загрузки модели вируса;
- 4 – кнопка загрузки модели бактерии.

Интерфейс AR-сцены приложения содержит в себе такие элементы, как:

- 1 – поле-подсказка, в котором отображается название и описание органеллы;
- 2 – пространство для отображения модели клетки;
- 3 – кнопка для просмотра анимации митоза (деления клетки);
- 4 – кнопка для просмотра реакции клетки на добавление воды;
- 5 – кнопка для просмотра питания клетки.

Интерфейс AR-сцены приложения представлен на рисунке 21.

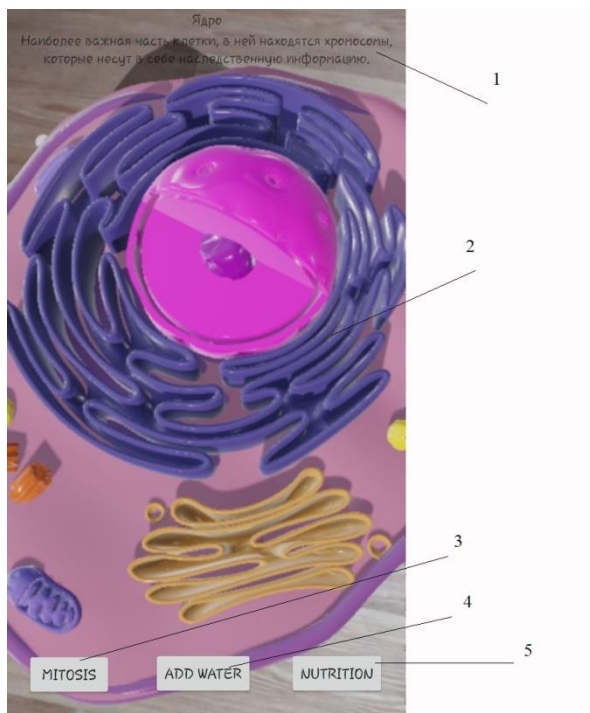


Рисунок 21 – Интерфейс AR-сцены приложения «Живая клетка»

Также перед входом в приложение пользователю выводится сообщение о том, что данное приложение использует технологию дополненной реальности. Данное сообщение представлено на рисунке 22.

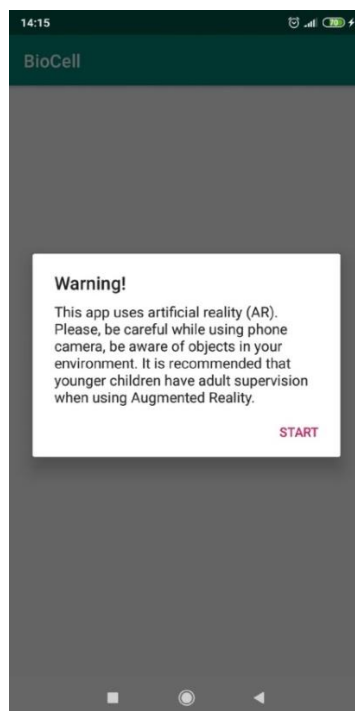


Рисунок 22 – Вывод сообщения-предупреждения для пользователя

Если у пользователя на устройстве нет приложения Experiment Starter и он не зарегистрирован в нем, то приложение выводит информирующее

сообщение, которое сообщает пользователю, что требуется сделать для использования приложения «Живая клетка». Данное сообщение изображено на рисунке 23.

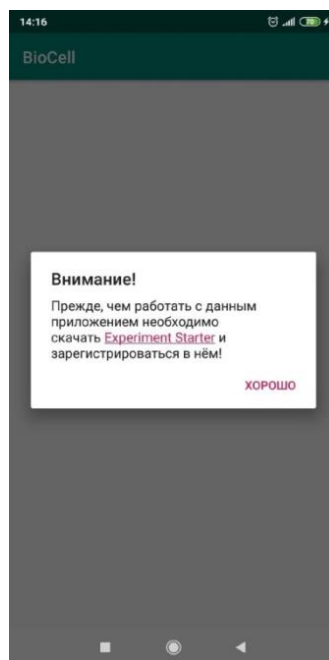


Рисунок 23 – Вывод сообщения для пользователя, не зарегистрированного в Experiment Starter

Если пользователь зарегистрирован в Experiment Starter, но не авторизован в нем, то пользователю выводится сообщение, представленное на рисунке 24.

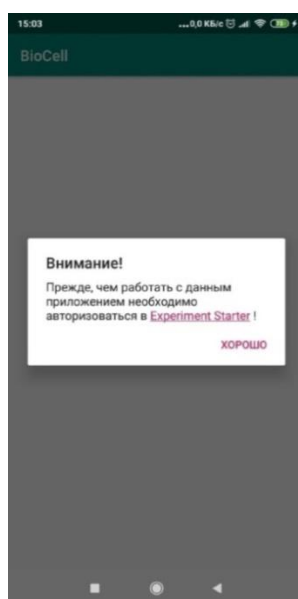


Рисунок 24 – Вывод сообщения о необходимости авторизации пользователя в Experiment Starter

При нажатии на одну из кнопок в главном меню, открывается AR-сцена, данная сцена работает с помощью камеры и приложению необходимо иметь доступ к камере. Для получения доступа к камере также выводится сообщение, запрашивающее его. Данное сообщение представлено на рисунке 25.

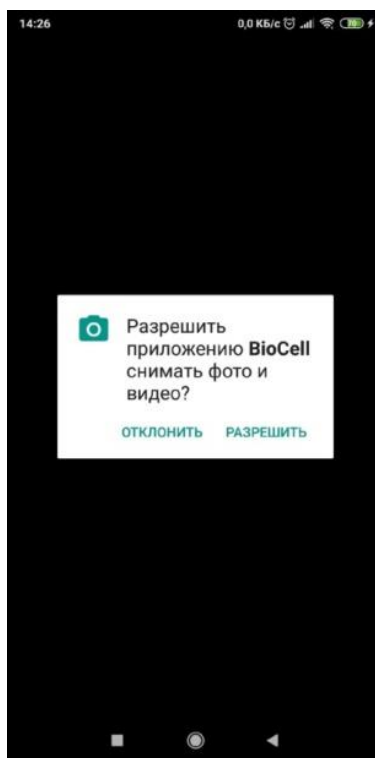


Рисунок 25 – Вывод сообщения о запросе на доступ приложения к камере

Интерфейс клиентской части приложения дополненной реальности «Живая клетка» прост и не требует большого количества времени для его изучения, а также понятен и удобен для любого пользователя.

3 Реализация подсистемы приложения дополненной реальности «Живая клетка»

3.1 Реализация пользовательского интерфейса

Интерфейс клиентской части приложения был реализован в среде Android Studio с помощью специального класса ViewGroup, который содержит дочерние классы, одним из них является ViewGroup.LayoutParams. ViewGroup.LayoutParams является базовым классом для макетов, используемых при разработке интерфейсов мобильных приложений.

На рисунке 26 показано, из каких элементов состоит данный интерфейс.

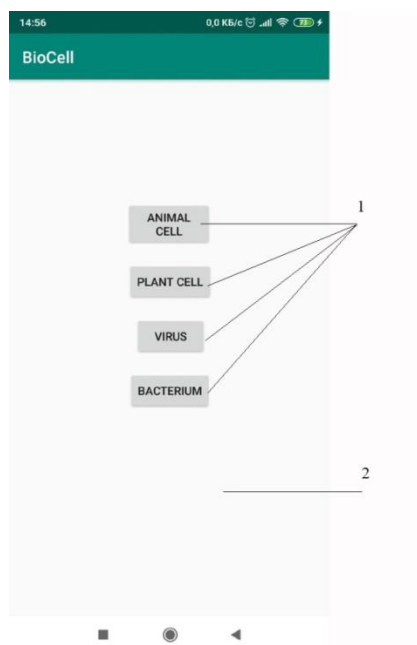


Рисунок 26 – Элементы интерфейса приложения «Живая клетка»

Для создания интерфейса были использованы такие элементы как:

- Button (кнопка);
- Constraint Layout.

Layout – это макет, который определяет взаимное расположение элементов на экране, структуру интерфейса, размещение этих элементов производится независимо от кода. Constraint Layout – макет, который позволяет привязывать элементы друг к другу, определять их взаимное расположение и расположение относительно самого Constraint Layout [7].

Данная компоновка удобна тем, что все элементы будут расположены в таком же соотношении вне зависимости от размеров экрана устройства.

Button – обычная кнопка, имеет несколько способов обработки нажатия на кнопку, в данном случае был использован атрибут onClick().

Интерфейс AR-сцены предполагает меньшее количество функций. Данный интерфейс приведен на рисунке 27.

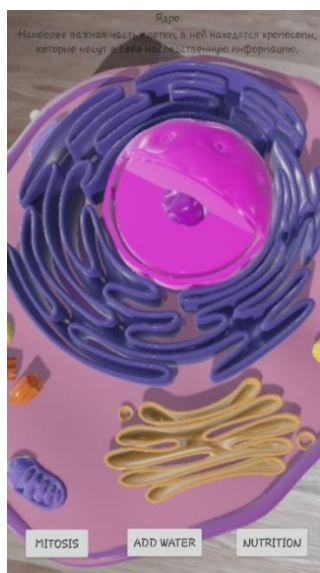


Рисунок 27 – Интерфейс AR-сцены приложения «Живая клетка»

Элементы, которые использовались при разработке данного интерфейса, показаны на рисунке 28.

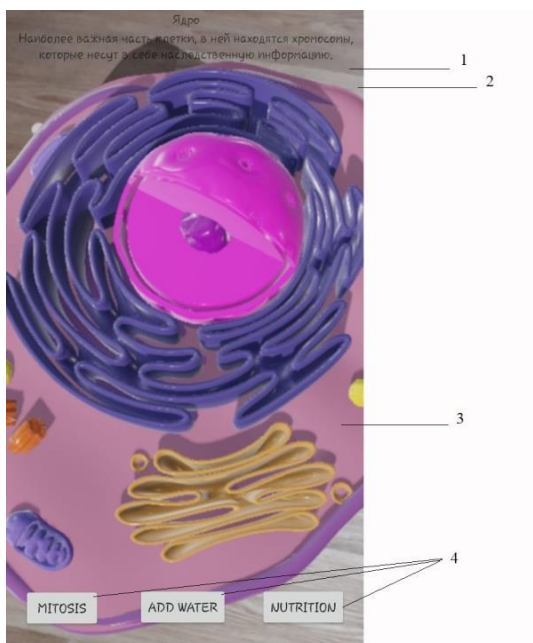


Рисунок 28 – Элементы интерфейса AR-сцены приложения «Живая клетка»

Для создания интерфейса AR-сцены были использованы следующие элементы:

- Relative Layout;
- <fragment>;
- TextView;
- Button.

Relative Layout – это макет, который позволяет располагать элементы относительного родительского элемента на макете.

<Fragment> является таким же объектом ViewGroup, как Button или TextView, но он имеет возможность самостоятельно обрабатывать события ввода и имеет свой жизненный цикл. В данном приложении <fragment> будет использован как ArFragment для создания AR-сцены.

Элемент TextView использован для создания подсказок. TextView позволяет отображать на экране текст, который пользователь не сможет отредактировать.

Разработка интерфейса была реализована с помощью средств Android Studio, которые позволяли, как располагать элементы интерфейса с помощью визуального конструктора, так и вручную настраивать взаиморасположение в xml-разметке.

3.2 Реализация программных средств

3.2.1 Диаграмма компонентов подсистемы приложения «Живая клетка»

Диаграмма компонентов позволяет определить физическую структуру программы, ее программные элементы и их взаимоотношения [3].

Также диаграмма компонентов позволяет установить зависимости между компонентами. В качестве компонентов выступают различные файлы, модули, библиотеки, пакеты. Для визуализации программной структуры приложения «Живая клетка» была построена диаграмма компонентов, изображенная на рисунке 29.

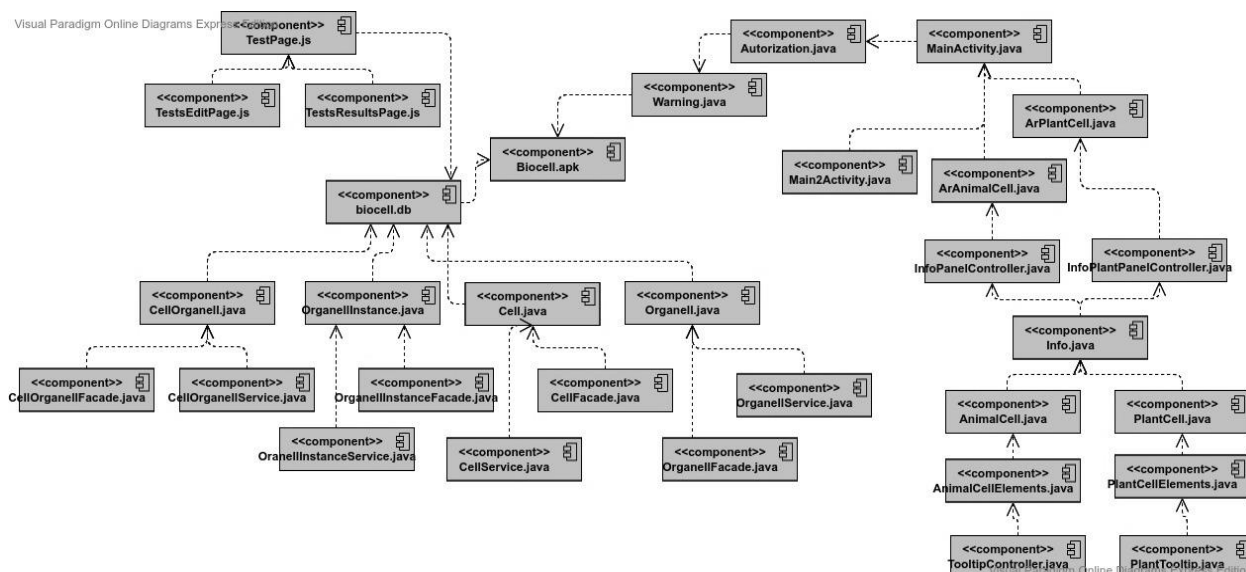


Рисунок 29 – Диаграмма компонентов приложения дополненной реальности
«Живая клетка»

Так как в данной выпускной квалификационной работе будет рассмотрена разработка подсистемы приложения дополненной реальности «Живая клетка», то следует показать те компоненты, реализация которых будет рассмотрена в данной работе. Диаграмма компонентов подсистемы приложения дополненной реальности «Живая клетка» будет представлена на рисунке 30.

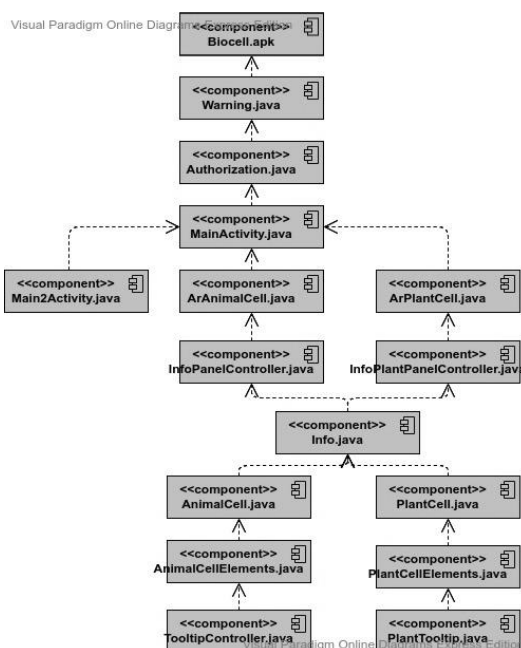


Рисунок 30 – Диаграмма компонентов подсистемы приложения дополненной реальности «Живая клетка»

Описание компонентов подсистемы приложения будет приведено ниже:

- Biocell.apk – исполняемый файл для операционной системы Android, который скачивается пользователем из Play Market;
- Warning.java – класс, в котором прописан вывод сообщения-предупреждения об использовании в данном приложении технологий дополненной реальности;
- Authorization.java – класс, отвечающий за проверку наличия у пользователя аккаунта Experiment Starter и вывод информирующих сообщений;
- MainActivity.java – класс, в котором прописана логика работы вызова моделей клеток и интерфейс меню;
- Main2Activity.java – класс, в котором прописан ArFragment, позволяющий загружать модели вируса и бактерии на Ar-сцену;
- ArAnimalCell.java – класс, в котором прописан ArFragment, позволяющий загружать модель животной клетки на Ar-сцену;
- ArPlantCell.java – класс, в котором прописан ArFragment, позволяющий загружать модель растительной клетки на Ar-сцену;
- InfoPanelController.java – класс, в котором происходит настройка работы подсказок для животной клетки;
- InfoPlantPanelController.java – класс, в котором происходит настройка работы подсказок для растительной клетки;
- Info.java – класс, в котором прописана структура подсказки.
- AnimalCell.java – класс, в котором настроен процесс сборки модели животной клетки из моделей органелл;
- PlantCell.java – класс, в котором настроен процесс сборки модели растительной клетки из моделей органелл;
- AnimalCellElements.java – класс, в котором описаны все органеллы животной клетки в соответствии с их именами, заданными в классе AnimalCell;

- PlantCellElements.java – класс, в котором описаны все органеллы растительной клетки в соответствии с их именами, заданными в классе PlantCell;
- TooltipController.java – класс, который содержит логику определения нужной подсказки для животной клетки;
- PlantTooltip.java – класс, который содержит логику определения нужной подсказки для растительной клетки.

3.2.2 Представление развертывания приложения «Живая клетка»

Для отображения инфраструктуры приложения и распределения ее компонентов по аппаратным узлам, а также отображения способа передачи информации между узлами, используется диаграмма развертывания. Диаграмма развертывания приложения дополненной реальности «Живая клетка» представлена на рисунке 31.

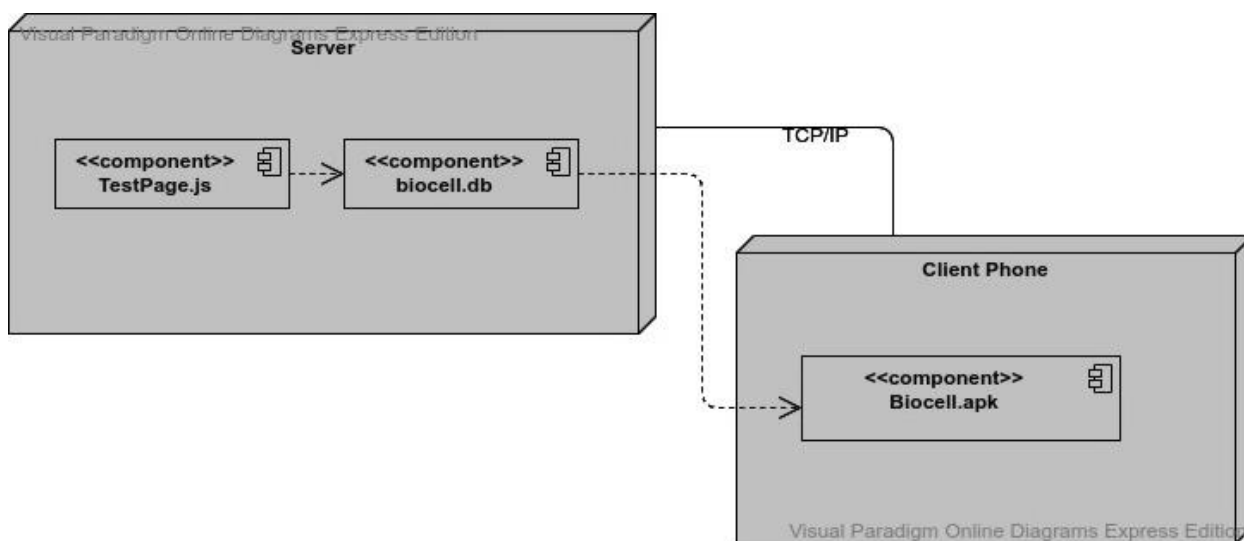


Рисунок 31 – Диаграмма развертывания приложения дополненной реальности «Живая клетка»

Данная диаграмма показывает, что клиентская часть приложения располагается на устройстве пользователя, а база данных приложения и сайт Experiment Starter располагаются на сервере.

3.2.3 Реализация процесса загрузки моделей клеток

Для приложения дополненной реальности «Живая клетка» была создана 3D-модель животной клетки. Моделирование клетки производилось в Blender, программе, позволяющей работать с трехмерными объектами. После создания формы клетки и ее органелл, на все составляющие клетки были наложены материалы (рисунок 32).

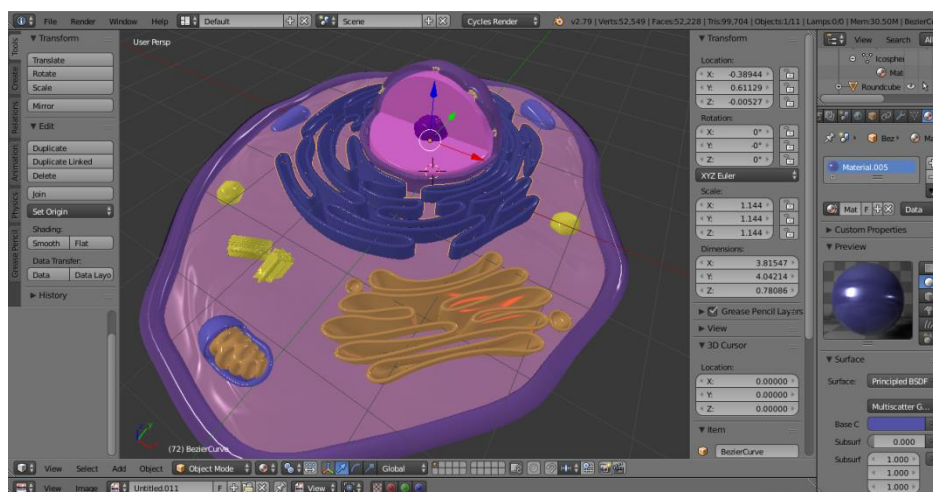


Рисунок 32 – Наложение материалов на составляющие клетки в Blender

Разработка приложения «Живая клетка» производилась с помощью средств разработки Sceneform SDK для ARCore от Google. Sceneform SDK – плагин, позволяющий создавать приложения с дополненной реальностью, AR-сцены для смартфонов и работать с 3D-объектами [8].

Для импорта в Android Studio, где производилась разработка приложения, 3D-модели из Blender были экспортированы в форматах .obj и .mtl [9]. После импорта в Android Studio, с помощью плагина Sceneform SDK модели были переведены в необходимые для дальнейшей работы форматы: .sfa и .sfb [10].

Для создания AR-сцены нужно использовать ArFragment, класс, который позволил бы работать с AR [11]. Для того, чтобы трехмерная модель не перемещалась относительно точки, где она была создана, использовался класс Anchor, этот класс позволяет «закрепить» виртуальный объект на одном месте [10]. Чтобы объект отображался и не перемещался без помощи пользователя, нужно расположить объект на координатной сетке и придать

ему определенную позу в пространстве, т. е. настроить ориентацию объекта в пространстве. Классы `ArFragment` и `Anchor` используются в методе `OnCreate()` класса `ArAnimalCell`, который приведен на рисунке 33.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main2);

    arFragment = (ArFragment) getSupportFragmentManager().findFragmentById(R.id.arFragment);
    mAnimalCell = new AnimalCell( arActivity: ArAnimalCell.this);
    arFragment.setOnTapArPlaneListener((hitResult, plane, motionEvent) ->{
        if(!isCreated) {
            Anchor anchor = hitResult.createAnchor();
            AnchorNode anchorNode = new AnchorNode(anchor);

            anchorNode.setParent(arFragment.getArSceneView().getScene());
            mAnimalCell.PlaceCell(anchorNode);
            isCreated = true;
        }
    });
}
```

Рисунок 33 – Метод `OnCreate()` класса `ArAnimalCell`

Модель клетки в приложении состоит из множества органелл, находящихся в ней. Соответственно, модели органелл были экспортированы в среду разработки отдельно. Это обусловлено тем, что анимацию процессов, происходящих в клетке гораздо проще проработать, если все органеллы будут загружены отдельно и будут собираться в единую модель.

Для отображения модели был использован класс `ModelRenderable`, который «привязывает» модель к элементу `Node`. `Node` позволяет преобразовывать модель в соответствии с иерархией сцены [7]. `Node` может содержать одного родителя и несколько дочерних элементов.

Построение модели клетки основано на использовании `Node`. Сначала был создан объект `TransformableNode`, который позволил бы масштабировать и перемещать модель по сцене, после него был создан `tmpNode` и родительским для него являлся объект класса `TransformableNode`. После этого, с помощью метода `setLocalRotation` было установлено вращение этого `Node` относительно родительского. Затем был создан `BaseNode`, родительским для которого являлся `tmpNode`. Для этого узла с помощью методов `setLocalPosition`, `setLocalRotation` и `setLocalScale`, были установлены позиция, вращение и размер относительно родительского `tmpNode`. Все

остальные объекты-органеллы являются дочерними для объекта BaseNode, и будут расположены относительно него. Фрагмент метода, в котором происходит сборка модели приведен на рисунке 34.

```
private void AssembleAnimalCell(){
    mCellElements = new AnimalCellElements();
    mCellElements.transformableBase = new TransformableNode(mArAnimalCell.getArFragment().getTransformationSystem());
    mCellElements.tmpNode = new Node();
    mCellElements.tmpNode.setParent(mCellElements.transformableBase);
    mCellElements.tmpNode.setLocalRotation(new Quaternion(new Vector3( v0: 0F, v1: 180F, v2: 0F)));
    mCellElements.BaseNode = new Node();
    mCellElements.BaseNode.setParent(mCellElements.tmpNode);
    mCellElements.BaseNode.setName("Cell");
    mCellElements.BaseNode.setLocalRotation(new Quaternion(new Vector3( v0: 0F, v1: -180F, v2: 0F)));
    mCellElements.BaseNode.setLocalPosition(new Vector3( v0: 0.0F, v1: -0.0F, v2: 0.0F));
    mCellElements.BaseNode.setLocalScale(new Vector3( v0: 0.1f, v1: 0.1f, v2: 0.1f));
    mCellElements.centrioli = new Node();
    mCellElements.centrioli.setParent(mCellElements.BaseNode);
    mCellElements.centrioli.setName("Centrioli");
    mCellElements.centrioli.setRenderable(mModels[0]);
    mCellElements.centrioli.setLocalPosition(new Vector3( v0: -1.0322F, v1: 2.488272F, v2: 2.63053F));
    mCellElements.centrioli.setWorldRotation(new Quaternion(new Vector3( v0: 0F, v1: 0F, v2: 0F)));
    mCellElements.centrioli.setLocalScale(new Vector3( v0: 1.192666f, v1: 1.192666f, v2: 1.192666f));
    mCellElements.citoplazma = new Node();//TransformableNode(mArFragment.getTransformationSystem());
    mCellElements.citoplazma.setParent(mCellElements.BaseNode);
    mCellElements.citoplazma.setName("Citoplazma");
    mCellElements.citoplazma.setRenderable(mModels[1]);
    mCellElements.citoplazma.setLocalPosition(new Vector3( v0: 0.197435F, v1: 2.588272F, v2: 0.415435F));
}
```

Рисунок 34 – Фрагмент метода AssembleAnimalCell()

Все органеллы хранятся в связном списке, и их визуализация происходит с помощью метода setRenderable(). Значения позиции и размера в соответствии с координатной сеткой были взяты такие же, какими они были установлены в Blender и были скорректированы, если в этом была необходимость.

Листинг кода клиентской части приложения приведен в приложении Б.

4 Тестирование подсистемы приложения дополненной реальности «Живая клетка»

4.1 Структурное тестирование

Структурное тестирование или тестирование по принципу белого ящика характеризуется тем, что разработчик теста имеет доступ к коду программы [12]. Данная методология тестирования предполагает выполнение каждого пути в программе.

Тестирование по принципу белого ящика включает в себя метод покрытия операторов, метод покрытия условий и метод покрытия маршрутов.

Тестирование может быть проведено на примере алгоритма «Загрузка модели клетки». Блок-схема алгоритма представлена на рисунке 35.

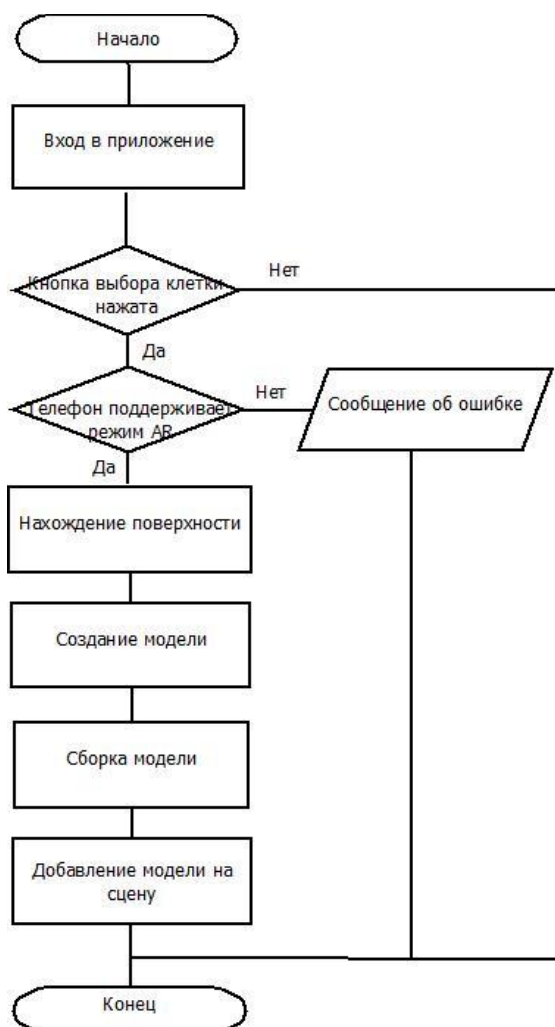


Рисунок 35 – Блок-схема алгоритма «Загрузка модели клетки»

Управляющий граф алгоритма, отображающий последовательность операторов, представлен на рисунке 36.

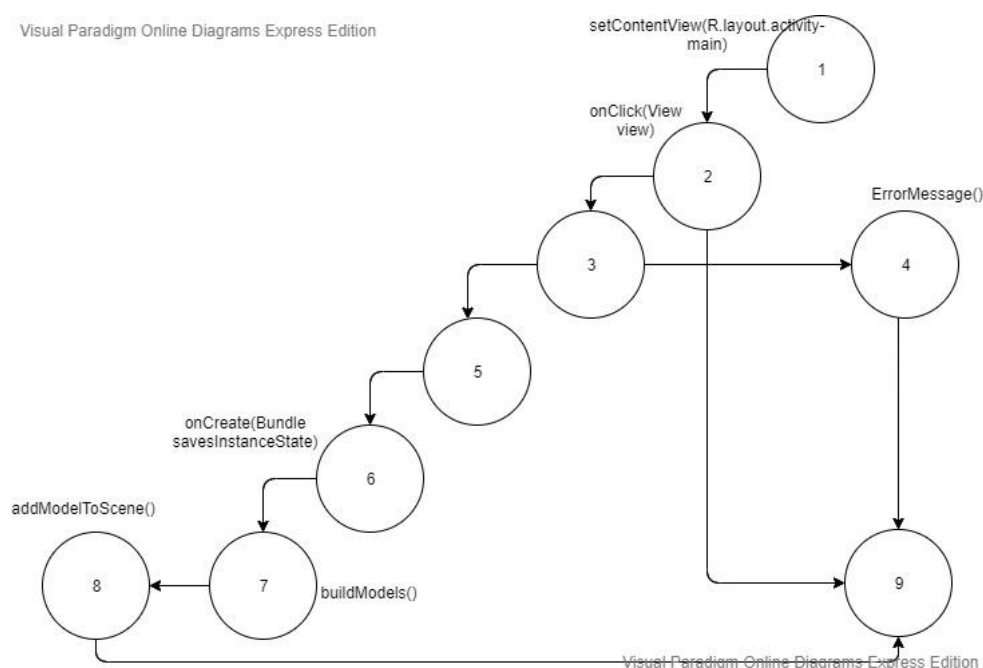


Рисунок 36 – Управляющий граф алгоритма «Загрузка модели клетки»

Управляющий граф позволяет отобразить поток управления программы.

4.1.1 Метод покрытия операторов

Критерии покрытия операторов подразумевает выполнение каждого оператора программы, по крайней мере, один раз.

Для выполнения каждого оператора было составлено 3 теста. Результаты приведены в таблице 13.

Таблица 13 – Результат тестирования методом покрытия операторов

Операторы	Тест	Фактический результат	Результат тестирования
1	Пользователь зашел в приложение, но не нажал ни на одну из кнопок.	Программа работает, но модели не загружены.	Успешно (рисунок В.1)

Продолжение таблицы 13

Операторы	Тест	Фактический результат	Результат тестирования
1, 2	Пользователь вошел в приложение, нажал на кнопку, его телефон не поддерживает режим дополненной	Сообщение о том, что устройство не	Успешно (рисунок В.2, В.3)
	реальности. Выводится сообщение о необходимости скачать сервисы Google Play AR, при переходе в Play Market, пользователю сообщается, что его устройство не поддерживает данный сервис.	поддерживает данный сервис	
1, 3, 4, 5, 6	Пользователь вошел в приложение, нажал на кнопку, его телефон поддерживает режим дополненной реальности, модель клетки загружается.	Модель клетки загружена	Успешно (рисунок В.4)

Все тесты прошли успешно. Каждый оператор был выполнен хотя бы один раз.

4.1.2 Метод покрытия условий

В методе покрытия условий нужно составить тесты, в которых все возможные результаты каждого условия в решении выполнялись хотя бы один раз. Также должно соблюдаться условие выполнения каждого оператора, по крайней мере, один раз.

Для выполнения каждого условия было составлено 3 теста. Результаты приведены в таблице 14.

Таблица 14 – Результат тестирования методом покрытия условий

Условия	Тест	Фактический результат	Результат тестирования
1 – нет	Пользователь вошел в приложение, но не нажимал на кнопку.	Приложение запущено, но модели не загружены	Успешно (рисунок В.1)

Продолжение таблицы 14

Условия	Тест	Фактический результат	Результат тестирования
1 – да, 2 – нет	Пользователь нажал кнопку, но его телефон не поддерживает режим дополненной реальности. Выводится сообщение о необходимости скачать сервисы Google Play AR, при переходе в PlayMarket, пользователю сообщается, что его устройство не поддерживает данный сервис.	Сообщение о том, что устройство не поддерживает данный сервис	Успешно (рисунок В.2, В.3)
1 – да, 2 – да	Пользователь нажал кнопку, его телефон поддерживает режим дополненной реальности, модель клетки загружается.	Модель клетки загружена	Успешно (рисунок В.4)

Все тесты прошли успешно. Каждое условие было выполнено хотя бы один раз.

4.1.3 Метод покрытия маршрутов

Метод покрытия маршрутов является наиболее полным и результативным. Он предполагает прохождение по каждому возможному пути программы. Для выполнения всех маршрутов было составлено 3 теста. Результаты приведены в таблице 15.

Таблица 15 – Результат тестирования методом покрытия маршрутов

Маршрут	Тест	Результат тестирования
1, 2, 9	Пользователь вошел в приложение, но не нажимал на кнопку.	Успешно (рисунок В.1)
1, 2, 3, 4, 9	Пользователь нажал кнопку, но его телефон не поддерживает режим AR. Выводится сообщение о необходимости скачать сервисы Google Play AR, при переходе в PlayMarket, пользователю сообщается, что его устройство не поддерживает данный сервис.	Успешно (рисунок В.2, В.3)
1, 2, 3, 5, 6, 7, 8, 9	Пользователь нажал кнопку, телефон поддерживает режим AR, модель загружается.	Успешно (рисунок В.4)

Все тесты прошли успешно. Каждый маршрут был пройден хотя бы один раз. Результаты структурного тестирования приведены в приложении В.

4.2 Анализ качества кода

4.2.1 Метрики кода

Метрики кода позволяют оценить качество и другие свойства определенного участка программного кода системы или приложения. Метрики кода являются совокупностью оценок ПО, служащей рекомендательными характеристиками и представляющую более полную информацию о программном коде [12].

Метрики кода в данной работе были рассмотрены на примере алгоритма загрузки модели клетки. Код всех методов представлен в приложении Б.

Метрики программного кода можно разделить на группы:

- метрики размера программ (Source Lines of Code - SLOC-метрика);
- метрики сложности потока управления программ (метрики Холстеда, цикломатическая сложность, метрики Чепина).

Вычислим соответствующие метрики кода на конкретных примерах.

4.2.2 SLOC-метрика

SLOC-метрика нужна для подсчета физических и логических строк кода. К физическим строкам относят все непустые строки исходного программного файла, к логическим строкам кода относят количество команд, использующихся в программе. Определение команды зависит от конкретного языка программирования.

Применим SLOC-метрику к программному коду для просмотра моделей клетки.

Результаты данной метрики представлены в таблице 16.

Таблица 16 – SLOC-метрики

Число физических строк	210
Число логических строк	193
Число пустых строк	0
Число строк-комментариев	28

Физические строки – это все строки кода, включая пустые и строки комментариев. Логические строки – это количество команд в коде, оно зависит от используемого языка программирования.

4.2.3 Цикломатическая сложность Маккейба

Цикломатическая сложность представляет счетное число линейно независимых маршрутов через программный код.

Граф потока управления – способ описания операций, хода программы с учетом управляющих конструкций и процедур [13]. Данный граф позволяет понять логику программы.

Для вычисления цикломатического числа Маккейба $Z(G)$ применяется формула:

$$Z(G) = e - v + 2p,$$

где e – число дуг ориентированного графа G ;

v – число вершин;

p – число компонентов связности графа.

Цикломатическое число Маккейба определяет количество тестовых прогонов программы, необходимых для тестирования по критерию «работает каждая ветвь».

Подсчет цикломатической сложности был проведён на основе управляющего графа алгоритма «Загрузка модели клеток» (смотри рисунок 36). Показатели для данного графа:

1) $e = 10$;

2) $v = 9$;

3) $p = 1$.

В результате вычисления цикломатического числа метрики Маккейба на основе данных показателей был получено значение 3. Так как

цикломатическая сложность метода не должна превышать 10, полученное значение является нормой.

4.2.4 Метрики Холстеда

Метрики Холстеда представлены четырьмя измеряемыми характеристиками программы:

- число уникальных операторов программы, включая символы-разделители, имена процедур и знаки операций;
- число уникальных операндов программы;
- общее число операторов в программе;
- общее число операндов в программе.

На основе метрик Холстеда можно получить следующие оценки:

- словарь программы;
- длина программы;
- объем программы;
- сложность программы;
- показатель усилий программиста.

Применим метрики Холстеда к нашему программному коду для загрузки модели клетки (приложение Б). Результаты данной метрики представлены в таблице 17.

Таблица 17 – Метрики Холстеда

Операторы	Число вхождений	Операнды	Число вхождений
=	41	@Override	1
.	267	modelRenderable	3
()	129	savedInstanceState	4
->	3	anchor	3
[]	10	anchorNode	6
==	1	this	2

Продолжение таблицы 17

Операторы	Число вхождений	Операнды	Число вхождений
onCreate()	4	intent	1
<>	2	mEngineElements	1
onClick()	1	id	1
:	6	mArActivity	2
<	1	arActivity	1
If()	1	arFragment	1
new	59	vector3	1
return	4	Uri.parse	2
		AnchorNode	1
		mModels	7
n1 = 14	N1 = 529	n2 = 16	N2 = 37

На основании этих характеристик рассчитываются следующие оценки.

Длина программы рассчитывается по формуле:

$$N = N1 + N2.$$

Словарь программы рассчитывается по формуле:

$$n = n1 + n2.$$

Вычислительная сложность рассчитывается по формуле:

$$HV = N * \log(2) n. \text{ (бит)}.$$

Под битом подразумевается логическая единица информации - символ, оператор, операнд. Объем программы по Холстеду определяет объем кода алгоритма. Размер метрики для одной функции должен находиться в пределах от 20 до 1000.

В результате вычисления измеряемых характеристик метрики Холстеда были получены следующие значения:

- 1) $N = 529$;
- 2) $n = 37$;

3) $HV = 2595,745$.

Полученный показатель объема программы по Холстеду находится в пределах нормы.

Такой показатель, как индекс сопровождаемости рассчитывается как метрика относительной сложности поддержки, но без слагаемого, учитывающего количество комментариев, и может принимать значения от 0 до 100:

Индекс сопровождаемости кода определяется по формуле:

$$MI = \max(0, (529 - 5.2 * \ln(HV) - 0.23 * CC - 16.2 * \ln(LoC)) * 100 / 529),$$

где HV (Halstead Volume) – вычислительная сложность;

CC (Cyclomatic Complexity) – структурная сложность кода, т.е. количество различных ветвей в коде;

LoC (Lines of Code) – количество строк кода.

$$CC = 3;$$

$$LOC = 210;$$

$$HV = 2595,745.$$

В результате вычисления получено значение индекса сопровождаемости кода, равное 82. Таким образом, алгоритм кода просмотр модели клеток имеет приемлемый индекс сопровождаемости.

5 Планирование и расчет бюджета разработки подсистемы приложения дополненной реальности «Живая клетка»

Управление проектом – это приложение знаний, опыта, методов и средств работы проекта для удовлетворения требований, предъявляемых к проекту, планирование, организация, контроль всех направлений проекта [14].

В качестве продукта управления проектами был выбран Microsoft Project, позволяющий оптимизировать планирование проекта, а также наглядно показать весь процесс разработки на сетевом графе и диаграмме Ганта. Для реализации планирования в среде Microsoft Project необходимо сначала определить список задач проекта [14]. Список задач проекта, их виды, связи, длительность и предшественники приведены в таблице 18.

Таблица 18 – Задачи проекта

№	Название	Вид задачи	Предшественник	Длительность
1	Начало реализации проекта	Веха	-	-
2	Анализ требований и постановка задачи	Фаза	-	6
3	Анализ программных продуктов аналогичного назначения	Задача	1	2
4	Постановка задачи	Задача	3	1
5	Анализ функциональных требований	Задача	3	2
6	Выбор средств разработки	Задача	4;5	2
7	Анализ требований и постановка задачи завершены	Веха	6	-
8	Проектирование приложения	Фаза	2	-
9	Функциональное моделирование	Задача	7	3
10	Проектирование структур данных	Задача	9	2
11	Проектирование программных средств	Задача	10	4

Продолжение таблицы 18

№	Название	Вид задачи	Предшественник	Длительность
12	Проектирование пользовательского интерфейса	Задача	10	2
13	Проектирование завершено	Веха	11;12	-
14	Реализация	Фаза	8	18
15	Реализация пользовательского интерфейса	Задача	13	3
16	Реализация программных средств	Задача	15	15
17	Реализация завершена	Веха	16	-
18	Отладка и тестирование	Фаза	14	8
19	Отладка клиентской части	Задача	17	3
20	Структурное тестирование	Задача	19	2
21	Рефакторинг	Задача	20	1
22	Анализ качества кода	Задача	21	2
23	Отладка и тестирование завершена	Веха	22	-
24	Завершение проекта	Фаза	18	6
25	Составление документации	Задача	23	5
26	Сдача в эксплуатацию	Задача	25	1
27	Проект завершен	Веха	26	-
28	Конец проекта	Веха	27	-

После определения состава работ и связей между ними требуется определить ресурсы, необходимые для выполнения задач проекта. Список ресурсов, их типов и соответствующих им затрат приведен в таблице 19.

Таблица 19 – Ресурсы и затраты проекта

№	Название ресурса	Тип ресурса	Затраты		Начисление
			Стандартная ставка	Ставка сверхур.	
1	Программист	Трудовой	20 000 р./мес.	250 р./ч.	Пропорциональное

Продолжение таблицы 19

№	Название ресурса	Тип ресурса	Затраты		Начисление
			Стандартная ставка	Ставка сверхур.	
2	Аналитик	Трудовой	14 000 р./мес.	150 р./ч.	Пропорциональное
3	Тестировщик	Трудовой	16 000 р./мес.	200 р./ч.	Пропорциональное
4	Бумага	Материальный	200 р.	-	Пропорциональное
5	Флэш-карта	Материальный	400 р.	-	Пропорциональное
6	Телефон с AR	Материальный	14 000р.	-	Пропорциональное
7	Компьютер	Материальный	28 000р.	-	Пропорциональное
8	Интернет	Затраты	-	-	Пропорциональное

Следующим шагом в планировании проекта является распределение ресурсов по задачам [15]. Распределение ресурсов по задачам приведено в таблице 20.

Таблица 20 – Распределение ресурсов проекта по задачам

№	Название	Ресурсы	Единицы (затраты)
1	Начало реализации проекта	Телефон с AR Компьютер	1 шт. 1 шт.
2	Анализ требований и постановка задачи		
3	Анализ программных продуктов аналогичного назначения	Интернет Аналитик	350р 100%
4	Постановка задачи	Программист	50%
5	Анализ функциональных требований	Программист Аналитик Интернет	50% 100% 350р
6	Выбор средств разработки	Интернет Программист	350р 100%
7	Анализ требований и постановка задачи завершены		
8	Проектирование приложений		
9	Функциональное моделирование	Программист	100%
10	Проектирование структур данных	Программист	100%
11	Проектирование программных средств	Программист	70%
12	Проектирование пользовательского интерфейса	Программист	30%
13	Проектирование завершено		

Продолжение таблицы 20

№	Название	Ресурсы	Единицы (затраты)
14	Реализация		
15	Реализация пользовательского интерфейса	Программист	100%
16	Реализация программных средств	Программист	100%
17	Реализация завершена		
18	Отладка и тестирование		
19	Отладка клиентской части	Программист Тестировщик	50% 100%
20	Структурное тестирование	Тестировщик	100%
21	Рефакторинг	Программист	100%
22	Анализ качества кода	Тестировщик Программист	100% 50%
23	Отладка и тестирование завершена		
24	Завершение проекта		
25	Составление документации	Бумага Аналитик Программист Тестировщик Флэш-карта Интернет	1уп. 100% 100% 100% 1шт. 350р
26	Сдача в эксплуатацию	Аналитик Программист Тестировщик	100% 100% 100%
27	Проект завершен		
28	Конец проекта		

В Microsoft Office Project для планирования процесса разработки подсистемы приложения дополненной реальности был создан проект, дата начала которого 13.04.20. В него были занесены проанализированные задачи, ресурсы и соответствующие им затраты. Результат добавления выделенных ресурсов (смотри таблицу 19) в проект приведен на рисунке 37.

		Название ресурса	Тип	Единицы измерения материалов	Краткое название	Группа	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Затраты на исполыз.	Начисление	Базовый календарь
1		Программист	Трудовой		П		100%	20 000,00 Р/мес	250,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
2		Аналитик	Трудовой		А		100%	14 000,00 Р/мес	150,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
3		Тестировщик	Трудовой		Т		100%	16 000,00 Р/мес	200,00 Р/ч	0,00 Р	Пропорциональное	Стандартный
4		Бумага	Материальный		Б			300,00 Р		0,00 Р	Пропорциональное	
5		Флэш-карта	Материальный		Ф			400,00 Р		0,00 Р	Пропорциональное	
6		Телефон с AR	Материальный		Т			14 000,00 Р		0,00 Р	Пропорциональное	
7		Компьютер	Материальный		К			28 000,00 Р		0,00 Р	Пропорциональное	
8		Интернет	Затраты		И						Пропорциональное	

Рисунок 37 – Лист ресурсов

После выполнения всех действий, Microsoft Project автоматически построил диаграмму Ганта, которая предлагает наглядное представление календарного плана с выделенными задачами и распределенными для них ресурсами. Диаграмма Ганта проекта приведена на рисунке 38.

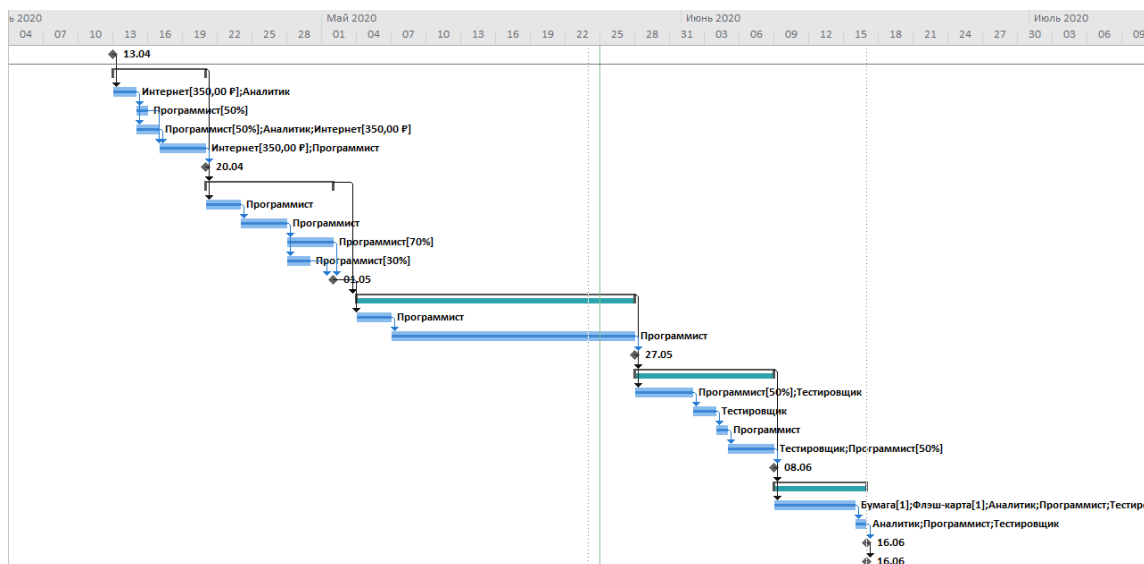


Рисунок 38 – Диаграмма Ганта разработанного проекта

В Microsoft Project был получен обзорный отчет планирования разработки подсистемы приложения дополненной реальности «Живая клетка», представленный на рисунке 39. В данном отчете приведены данные о дате начала и окончания проекта, его длительности, проценте завершения проекта и проценте завершения всех задач верхнего уровня.

ОБЗОР ПРОЕКТА

ПН 13.04.20 - ВТ 16.06.20 47 ДНЕЙ

% ЗАВЕРШЕНИЯ

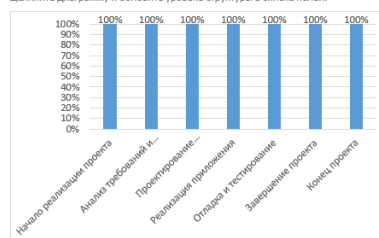
100%

ВЕЩИ С НАСТУПИВШИМ СРОКОМ
Приближающиеся веки

Название	Описание

% ЗАВЕРШЕНИЯ

Состояние всех задач верхнего уровня. Чтобы просмотреть состояние вложенных задач, щелкните диаграмму и обновите уровень структуры в списке полей.



ЗАДАЧИ С ЗАДЕРЖКОЙ

Простроченные задачи

Название	Начало	Окончание	Длительность	% завершения	Названия ресурсов
----------	--------	-----------	--------------	--------------	-------------------

Рисунок 39 – Обзорный отчет планирования разработки приложения

Также Microsoft Project есть возможность построить отчет по движению денежных средств. Отчет по движению денежных средств представлен на рисунке 40.

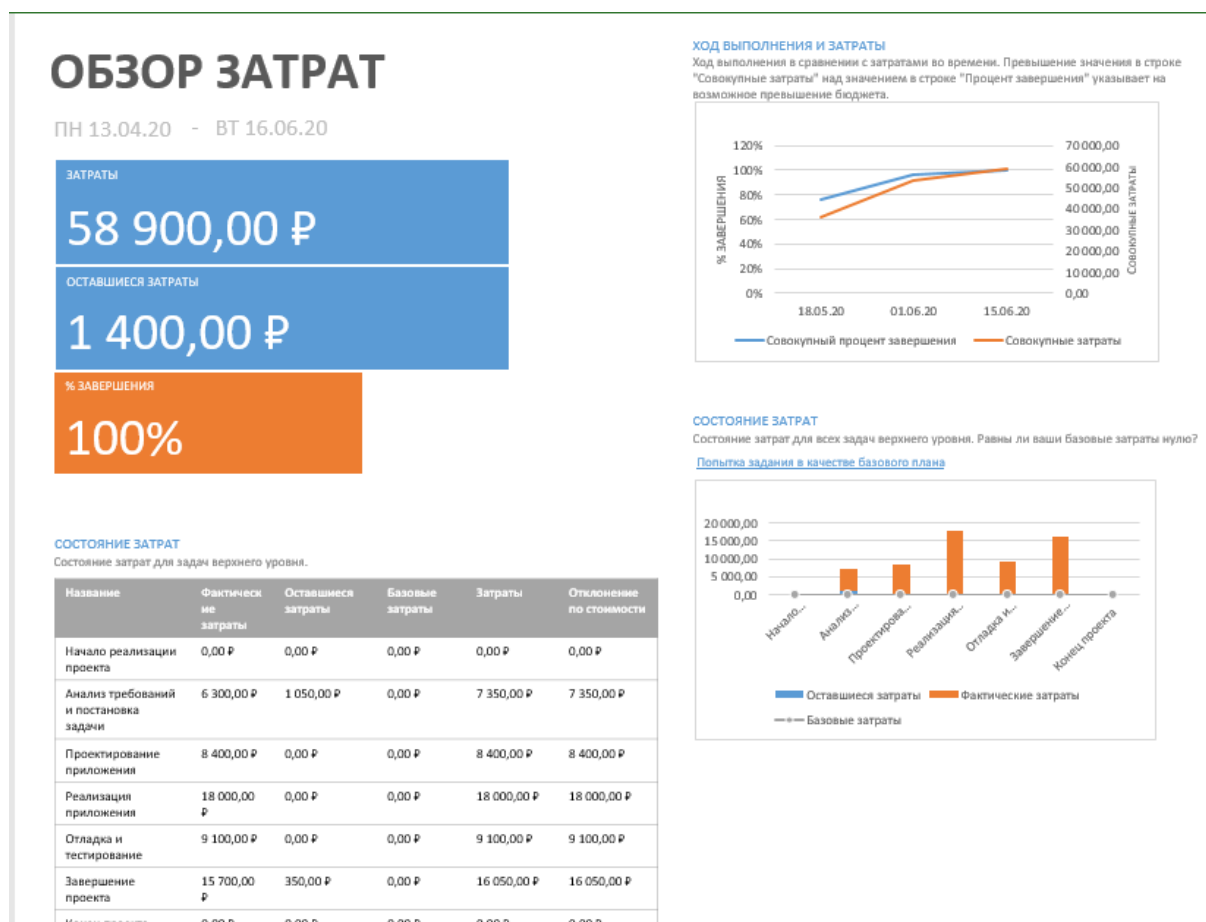


Рисунок 40 – Отчет по движению денежных средств

В данном отчете отображаются общие затраты проекта, процент завершения проекта, затраты для каждой задачи верхнего уровня, а также ход выполнения проекта в соответствии с затратами во времени. По данному отчету можно увидеть, что самой затратной задачей оказалась реализация приложения, которая составила 18 000 рублей.

Также Microsoft Project позволяет построить обзорный отчет трудозатрат, который показывает выработку трудозатрат, фактические трудовые затраты и оставшиеся затраты в конце проекта. Обзорный отчет по трудозатратам приведен на рисунке 41.

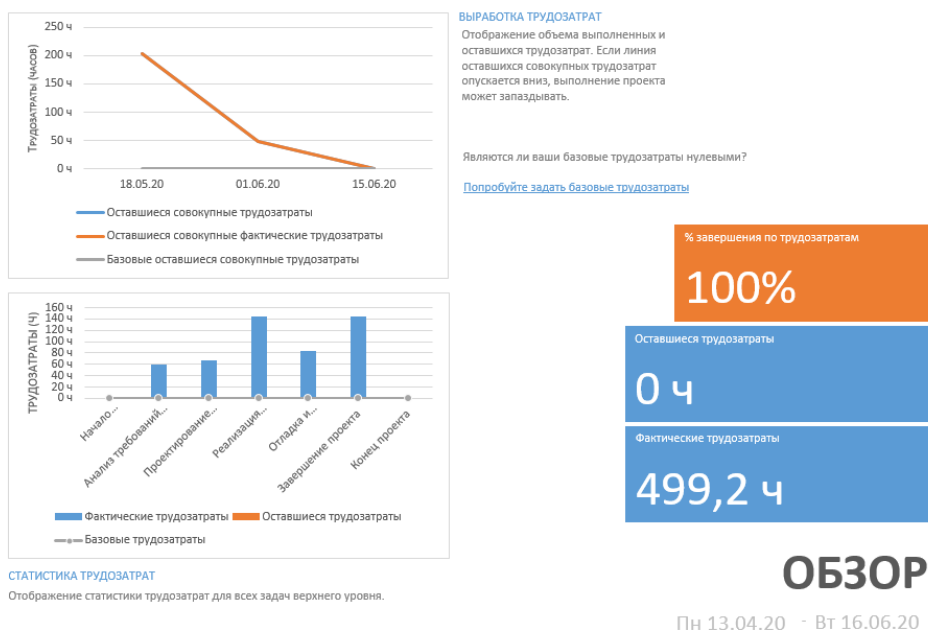


Рисунок 41 – Обзорный отчет по трудозатратам

По результатам приведенных отчетов видно, что срок выполнения проекта составил 47 дней, с учетом праздников и выходных дней, а бюджет проекта составил 58 900 рублей. Благодаря оценке сроков и бюджета проекта можно сделать вывод о том, что процесс разработки подсистемы приложения дополненной реальности «Живая клетка» является экономически выгодным. Данный программный продукт за счет низкой стоимости и достаточно быстрой разработки станет конкурентом для существующих аналогичных систем.

Заключение

В результате выпускной квалификационной работы, была разработана подсистема приложения дополненной реальности «Живая клетка».

Был проведен анализ программных аналогов на рынке Google Play, проведено проектирование интерфейса, структур данных и программных средств приложения. Средством разработки была выбрана среда Android Studio, а языком программирования Java. Также было проведено структурное тестирование и анализ качества кода приложения.

Функциональное моделирование бизнес-процессов приложения было проведено с использованием методологий IDEF0, DFD, IDEF3. Проектирование структур данных приложения дополненной реальности было проведено с использованием методологии IDEF1X. Проектирование программных средств было произведено с помощью унифицированного языка моделирования UML.

Планирование и расчет бюджета приложения были произведены в программе Microsoft Project. На основе выделенных задач и ресурсов проекта была построена диаграмма Ганта и проведен расчет бюджета проекта. Срок выполнения проекта составил 47 дней, бюджет проекта – 58 900 рублей.

Разработанное приложение внедрено в образовательную платформу Experiment, через которую осуществляется вход в приложение. Приложение используется для изучения строения микроорганизмов. Акт о результатах внедрения приложения в образовательную платформу Experiment приведен в приложении Г.

Список использованных источников

1. Полат, Е.С. Новые педагогические и информационные технологии в системе образования: учебное пособие / Под ред. Е.С. Полат. – М.: Академия, 2001. – 272 с.
2. Шилдт, Г. Java. Полное руководство, 10-е изд. / Пер. с англ. – СПб.: ООО «Альфа-книга», 2018. – 1488 с.
3. Якобсон, А. Унифицированный процесс разработки программного обеспечения / А. Якобсон, Г. Буч, Дж. Рамбо. – СПб.: Питер. – 2002. – 496 с.
4. Мюллер, Роберт Дж. Базы данных и UML: Проектирование / Роберт Дж. Мюллер – М.: ЛОРИ, 2002 – 420 с.
5. Гульяев, А.К. Проектирование и дизайн пользовательского интерфейса / А.К. Гульяев, В.А. Машин: учебное пособие. – СПб.: Корона-Принт, 2007. – 239 с.
6. Айрапетян, Г.М. Дизайн мобильного приложения / Г. М. Айрапетян // Молодой учёный. – 2018. – №48(234). – С.12 – 15.
7. Руководство Android-разработчика // Сервисы Google для мобильных разработчиков [Электронный ресурс]. URL: <https://developer.android.com/guide> (дата обращения 10.06.2020).
8. Майер, Р. Программирование приложений для планшетных компьютеров и смартфонов / Рето Майер. – М.: Эксмо, 2013. – 816 с.
9. Като, Х. Манипуляция виртуальными объектами в настольной среде AR / Х. Като, М. Биллингхерст, И. Поупырев, К. Имамото, К. Тачибана // ISAR'00. – 2000. – С.111-119.
10. Машнин, Т. Разработка Android-приложений с Augmented Reality / Т. Машнин. – М.: ЛитРес, 2017. – 250 с.
11. Дарвин, Ян Ф. Android Сборник рецептов: задачи и решения для разработчиков приложений, 2-е изд./ Пер. с англ. – СПб. : ООО «Альфа-книга», 2018. – 768 с.

12. Виды тестирования программного обеспечения // Тестирование программного обеспечения [Электронный ресурс]. – URL: <http://www.protesting.ru/testing/testtypes.html> (дата обращения: 06.05.2020).

13. Майерс, Г. Искусство тестирования программ / Пер. с англ. под ред. Б.А. Позина. – М. : Финансы и статистика, 1982. – 176 с.

14. Четфилд, К. Microsoft Project 2016 / Четфилд К., Джонсон Т. – М.: ЭКОМ Паблишерз, 2018. – 594 с.

15. Куперштейн, В. И. Microsoft Project в управлении проектами. / В. И. Куперштейн, А.В. Цветкова. – СПб.: БХВ-Петербург, 2011. – 415 с.

Приложение А

ГЛОССАРИЙ

(справочное)

Дополненная реальность (augmented reality, AR) – это среда, в реальном времени дополняющая физический мир, каким мы его видим, цифровыми данными с помощью каких-либо устройств – планшетов, смартфонов или других, и программной части.

Android Studio – это интегрированная среда разработки для работы с платформой Android.

Программное обеспечение (ПО) – это программа или множество программ, используемых для управления компьютером.

База данных (БД) - это организованная структура, предназначенная для хранения, изменения и обработки взаимосвязанной информации, преимущественно больших объемов.

Системы управления базами данных (СУБД) – это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Play Маркет (Google Play, Android Market) – магазин приложений, игр, книг, музыки и фильмов компании Google и других компаний, позволяющий владельцам устройств с операционной системой Android устанавливать и приобретать различные приложения.

Blender – профессиональное открытое программное обеспечение для создания трёхмерной компьютерной графики, включающее в себя средства моделирования, анимации, рендеринга, постобработки и монтажа видео со звуком, компоновки с помощью «узлов», а также для создания интерактивных игр.

Приложение Б

ЛИСТИНГ ПРОГРАММЫ НА ЯЗЫКЕ JAVA

(рекомендуемое)

Листинг Б.1 – Код класса MainActivity

```

public class MainActivity extends AppCompatActivity {
    // private TextView get_response_text;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void onClick(View view) {
        String obj = "";
        switch(view.getId()){
            case R.id.bact:
                obj = "bacteriyaa.sfb";
                Intent intent = new Intent(MainActivity.this,
Main2Activity.class);
                intent.putExtra("Source", obj);
                startActivity(intent);
                break;
            case R.id.animalcell:
                Intent intent1 = new
Intent(MainActivity.this,ArAnimalCell.class);
                //intent.putExtra("Source", obj);
                startActivity(intent1);
                break;
            case R.id.plantcell:
                Intent intent2 = new Intent(MainActivity.this,
ArPlantCell.class);
                //intent.putExtra("Source", obj);
                startActivity(intent2);
                break;
            case R.id.virus: obj = "virus_razd.sfb";
                Intent intent3 = new Intent(MainActivity.this,
Main2Activity.class);
                intent3.putExtra("Source", obj);
                startActivity(intent3);break;
        }
    }
}

```

Листинг Б.2 – Код класса ArAnimalCell

```

public class ArAnimalCell extends AppCompatActivity {
    private ArFragment arFragment;
    public ArFragment getArFragment(){return arFragment;}

    private AnimalCell mAnimalCell;
    public AnimalCell getAnimalCell(){return mAnimalCell;}
    private boolean isCreated = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        arFragment = (ArFragment)
getSupportFragmentManager().findFragmentById(R.id.arFragment);
    }
}

```

```

        mAnimalCell = new AnimalCell(ArAnimalCell.this);
        arFragment.setOnTapArPlaneListener((hitResult, plane,
motionEvent) ->{
            if(!isCreated) {
                Anchor anchor = hitResult.createAnchor();
                AnchorNode anchorNode = new AnchorNode(anchor);

                anchorNode.setParent(arFragment.getArSceneView().getScene());
                mAnimalCell.PlaceCell(anchorNode);
                isCreated = true;
            }
        });
    }
    public void OnCloseInfoPanel(View view){
        mAnimalCell.getToltipController().getPanelController().Disactivate();
    }
}

```

Листинг Б.3 – Код класса AnimalCell

```

public class AnimalCell {

    //Ссылка на текущую активити
    private ArAnimalCell mArAnimalCell;

    //Контроллер системы подсказок
    private ToltipController mToltipController;
    public ToltipController getToltipController(){
        return mToltipController;
    }
    //Элементы
    private AnimalCellElements mCellElements;
    public AnimalCellElements getEngineElements(){return
mCellElements;}
    //Собранные модели ассетов
    private ModelRendererable[] mModels;

    private LinkedList<Node> mTransparentGameobjects;
    //Конструктор
    public AnimalCell(ArAnimalCell arActivity){
        mArAnimalCell = arActivity;
        this.BuildModels();
        mTransparentGameobjects = new LinkedList<>();
    }
    //Разместить клетку на AnchorNode
    public void PlaceCell(AnchorNode anchorNode){
        if(mCellElements == null)
            this.AssembleAnimalCell();
        mCellElements.transformableBase.setParent(anchorNode);
    }

    private boolean mIsWorking = true;

    public boolean getIsWorking() {
        return mIsWorking;
    }
}

```

```

    }
    //Собрать модели из ассетов
    private void BuildModels(){
        final int count = 11;
        final ModelRenderer[] models = new
ModelRenderable[count];
        final String[] uris = new String[]{
            "centrioli.sfb",
            "citoplazma.sfb",
            "endset.sfb",
            "goldzhi.sfb",
            "lizosomi.sfb",
            "membrana.sfb",
            "mitoh1.sfb",
            "mitoh2.sfb",
            "mitoh3.sfb",
            "yadrishko.sfb",
            "yadro.sfb",

        };

        for(int i = 0; i<count; i++){
            int finalI = i;
            ModelRenderer
                .builder()
                .setSource(mArAnimalCell, Uri.parse(uris[i]))
                .build()
                .thenAccept(renderable -> models[finalI] =
renderable)

                .exceptionally(throwable -> {
                    Toast toast =
                        Toast.makeText(mArAnimalCell, "Unable
to load renderable", Toast.LENGTH_LONG);
                    toast.show();
                    return null;
                });
        }

        mModels = models;
    }

    //Собрать комплексную модель двигателя
    private void AssembleAnimalCell(){

        mCellElements = new AnimalCellElements();

        mCellElements.transformableBase = new
TransformableNode(mArAnimalCell.getArFragment().getTransformationSyste
m());

        mCellElements.tmpNode = new Node();

        mCellElements.tmpNode.setParent(mCellElements.transformableBase);
        mCellElements.tmpNode.setLocalRotation(new Quaternion(new
Vector3(0F,180F,0F)));

        mCellElements.BaseNode = new Node();
        mCellElements.BaseNode.setParent(mCellElements.tmpNode);
        mCellElements.BaseNode.setName("Cell");
    }

```

```

        mCellElements.BaseNode.setLocalRotation(new
Quaternion(new Vector3(0F,-180F,0F)));
        mCellElements.BaseNode.setLocalPosition(new
Vector3(0.0F,-0.0F,0.0F));
        mCellElements.BaseNode.setLocalScale(new
Vector3(0.1f,0.1f,0.1f));
        mCellElements.centrioli = new Node();

mCellElements.centrioli.setParent(mCellElements.BaseNode);
        mCellElements.centrioli.setName("Centrioli");
        mCellElements.centrioli.setRenderable(mModels[0]);
        mCellElements.centrioli.setLocalPosition(new Vector3(-
1.0322F,2.488272F,2.63053F));
        mCellElements.centrioli.setWorldRotation(new
Quaternion(new Vector3(0F,0F,0F)));
        mCellElements.centrioli.setLocalScale(new
Vector3(1.192666f,1.192666f,1.192666f));

        mCellElements.citoplazma = new
Node();//TransformableNode(mArFragment.getTransformationSystem());

mCellElements.citoplazma.setParent(mCellElements.BaseNode);
        mCellElements.citoplazma.setName("Citoplazma");
        mCellElements.citoplazma.setRenderable(mModels[1]);
        mCellElements.citoplazma.setLocalPosition(new
Vector3(0.197435F,2.588272F,0.415435F));
        mCellElements.citoplazma.setLocalRotation(new
Quaternion(new Vector3(0F,-0F,0F)));
        mCellElements.citoplazma.setLocalScale(new
Vector3(1.10056f,1.10056f,1.12878f));

        mCellElements.endset = new Node();
        mCellElements.endset.setParent(mCellElements.BaseNode);
        mCellElements.endset.setName("Endset");
        mCellElements.endset.setRenderable(mModels[2]);
        mCellElements.endset.setLocalPosition(new Vector3(
0.0F,2.588272F,-0.400271F));
        mCellElements.endset.setLocalRotation(new Quaternion(new
Vector3(0F,-0F,0F))); //x: -90
        mCellElements.endset.setLocalScale(new
Vector3(1.14412f,1.14412f,1.14412f));

        mCellElements.goldzhi= new Node();
        mCellElements.goldzhi.setParent(mCellElements.BaseNode);
        mCellElements.goldzhi.setName("Complex Goldzhi");
        mCellElements.goldzhi.setRenderable(mModels[3]);
        mCellElements.goldzhi.setLocalPosition(new Vector3(
1.79929F,2.088272F,1.98075F));
        mCellElements.goldzhi.setLocalScale(new
Vector3(1.100696f,1.100696f,1.100696f));
        mCellElements.goldzhi.setLocalRotation(new Quaternion(new
Vector3(0F,0F,0.0F)));

        mCellElements.lizosomi = new Node();
        mCellElements.lizosomi.setName("Lizosoma");
        mCellElements.lizosomi.setParent(mCellElements.BaseNode);
        mCellElements.lizosomi.setRenderable(mModels[4]);
        mCellElements.lizosomi.setLocalPosition(new
Vector3(0.55696F,2.488272F,1.108906F));

```

```

        mCellElements.lizosomi.setLocalRotation(new
Quaternion(new Vector3(0.0F,0.0F,0.0F)));
        //mEngineElements.lizosomi.setLocalScale(new
Vector3(0.192666f,0.192666f,0.192666f));

        mCellElements.membrana = new Node();
        mCellElements.membrana.setName("Membrana");
        mCellElements.membrana.setRenderable(mModels[5]);
        mCellElements.membrana.setParent(mCellElements.BaseNode);
        mCellElements.membrana.setLocalPosition(new
Vector3(0.202059F,-0.126908F,0.507435F));
        mCellElements.membrana.setLocalScale(new
Vector3(3.20168f,3.00168f,3.28634f));
        mCellElements.membrana.setLocalRotation(new
Quaternion(new Vector3(0F,-0F,0F)));
        mCellElements.mitoh1 = new Node();
        mCellElements.mitoh1.setName("Mitochondria1");
        mCellElements.mitoh1.setParent(mCellElements.BaseNode);
        mCellElements.mitoh1.setRenderable(mModels[6]);
        mCellElements.mitoh1.setLocalPosition(new
Vector3(0.046483F,2.588272F,3.614486F));
        mCellElements.mitoh1.setLocalRotation(new Quaternion(new
Vector3(0,0,0)));
        mCellElements.mitoh1.setLocalScale(new
Vector3(1.045119f,1.052492f,1.045119f));

        mCellElements.mitoh2 = new Node();
        mCellElements.mitoh2.setName("Mitochondria2");
        mCellElements.mitoh2.setRenderable(mModels[7]);
        mCellElements.mitoh2.setLocalPosition(new
Vector3(1.880461F,2.588272F,-1.954486F));
        mCellElements.mitoh2.setParent(mCellElements.BaseNode);
        mCellElements.mitoh2.setLocalRotation(new Quaternion(new
Vector3(0F,0F,0F)));

        mCellElements.mitoh3 = new Node();
        mCellElements.mitoh3.setName("Mitochondria3");
        mCellElements.mitoh3.setParent(mCellElements.BaseNode);
        mCellElements.mitoh3.setRenderable(mModels[8]);
        mCellElements.mitoh3.setLocalPosition(new Vector3(-
2.405461F,2.588272F,-0.724486F));
        mCellElements.mitoh3.setLocalRotation(new Quaternion(new
Vector3(0,0,0)));

        mCellElements.yadrishko = new Node();
        mCellElements.yadrishko.setName("Yadrishko");
        mCellElements.yadrishko.setRenderable(mModels[9]);

        mCellElements.yadrishko.setParent(mCellElements.BaseNode);
        mCellElements.yadrishko.setLocalPosition(new
Vector3(0.077435F,3.188272F,-0.615435F));
        mCellElements.yadrishko.setLocalScale(new
Vector3(1.357931f,1.357931f,1.357931f));
        mCellElements.yadrishko.setLocalRotation(new
Quaternion(new Vector3(0F,-0F,0F)));

        mCellElements.yadro = new Node();
        mCellElements.yadro.setName("Yadro");
        mCellElements.yadro.setRenderable(mModels[10]);

```

```

        mCellElements.yadro.setParent(mCellElements.BaseNode);
        mCellElements.yadro.setLocalPosition(new
Vector3(0.077435F, 2.288272F, -0.615435F));
        mCellElements.yadro.setLocalRotation(new Quaternion(new
Vector3(0, -0, 0)));
        mCellElements.yadro.setLocalScale(new
Vector3(2.24412f, 2.24412f, 2.24412f));
        mTooltipController = new
TooltipController(mArAnimalCell, mCellElements);
        mTooltipController.setParent(mCellElements.BaseNode);
    }
}

```

Листинг Б.4 – Код класса AnimalCellElements

```

public class AnimalCellElements {
    private AnimalCell mAnimalCell;

    public TransformableNode transformableBase;

    //Объекты
    public Node
        tmpNode,
        BaseNode,
        yadro,
        yadrishko,
        mitoh1,
        mitoh2,
        mitoh3,
        membrana,
        lizosomi,
        goldzhi,
        endset,
        citoplazma,
        centrioli;

    private static Info[] partInfoData = new Info[]{
        new Info("Yadro",
            "Ядро",
            "Наиболее важная часть клетки, в ней находятся
хромосомы, которые несут в себе наследственную информацию."),
        new Info("Yadrishko",
            "Ядрышко",
            "Образует рибосомные субъединицы"),
        new Info("Mitochondrial",
            "Митохондрия",
            "Энергитическая станция клетки"),
        new Info("Mitochondria2",
            "Митохондрия",
            "Энергитическая станция клетки"),
        new Info("Mitochondria3",
            "Митохондрия",
            "Энергитическая станция клетки"),
    };
}

```

```

        new Info("Membrana",
            "Мембрана",
            "Защищает клетку от внешних воздействий и
            отвечает за попадание в клетку питательных веществ"),

        new Info("Lizosoma",
            "Лизосома",
            "Отвечает за клеточное пищеварение,
            уничтожение ненужных клетке органоидов, саморазрушение клетки,
            секрецию веществ за пределы клетки."),

        new Info("Complex Goldzhi",
            "Комплекс Гольджи",
            "Переносит и преобразует белки, производит
            липиды"),

        new Info("Endset",
            "Эндоплазматическая сеть",
            "Отвечает за синтез белков, липидов и
            углеводов, а также участвует в транспорте веществ в клетке"),

        new Info("Citoplazma",
            "Цитоплазма",
            "Объединяет все органеллы и обеспечивает их
            взаимодействие"),

        new Info("Centrioli",
            "Центриоли",
            "Образуют клеточный центр, который нужен при
            делении клетки")

    };

    public static Info FindPartInfo(String nodeName){
        Info result = null;
        for (Info pInfo : partInfoData) {
            if(pInfo.nodeName == nodeName){
                result = pInfo;
            }
        }
        return result;
    }
}

```

Листинг Б.5 – Код класса TooltipController

```

public class TooltipController extends Node {
    private ArAnimalCell mArAnimalCell;
    //Ссылка на структуру объектов
    private AnimalCellElements mElements;
    private InfoPanelController mPanelController;
    public InfoPanelController getPanelController() {
        return mPanelController;
    };
    //Объект на которые указали
    private Node mTargetNode;
    //Точка в пространстве, полученная в результате выделения объекта

```

```

private Vector3 mTargetPoint;

//Высрывающаяся подсказка
private TextView mTextView;

//Счётчик кадров
private int mFrameCounter = 0;
private int mTextViewHeight;
private int mTextViewWidth;

// Layout в котором находится подсказка (TextView)
private RelativeLayout
    mLayout;
//Ar core камера
private Camera mCamera;
//Параметры создания объекта (TextView)
private ViewGroup.LayoutParams mLayoutParams;

//Padding в формате DIP и переведённый в PX для данного устройства
private final float paddingInDIP = 6;
private int paddingInPX;

@SuppressLint("ResourceType")
public TooltipController(ArAnimalCell activity, AnimalCellElements
elements) {
    mArAnimalCell = activity;
    mElements = elements;

    mLayout = mArAnimalCell.findViewById(R.id.c_layout);
    OnTouchListener nodeTouchListener = new OnTouchListener() {
        @Override
        public boolean onTouch(HitTestResult hitTestResult,
MotionEvent motionEvent) {
            switch (motionEvent.getAction()) {
                case MotionEvent.ACTION_DOWN:
                    InitCheckCast(hitTestResult);
                    break;
                case MotionEvent.ACTION_HOVER_MOVE:
                    mLayout.removeView(mTextView);
                    mTextView = null;
                    break;
            }
            return false;
        }
    };
    mElements.BaseNode.setOnTouchListener(nodeTouchListener);

    mArAnimalCell.getArFragment().getArSceneView().setOnTouchListener(view
OnTouchListener); //TODO плохо работает
    mCamera =
    mArAnimalCell.getArFragment().getArSceneView().getScene().getCamera();
    mLayoutParams = new RelativeLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT,
        ViewGroup.LayoutParams.WRAP_CONTENT);

    //Перевод из DIP в PX
    Resources r = mArAnimalCell.getResources();
    paddingInPX = (int) TypedValue.applyDimension(
        TypedValue.COMPLEX_UNIT_DIP,

```

```

        paddingInDIP,
        r.getDisplayMetrics());

    mPanelController = new InfoPanelController(mArAnimalCell);
}

// Инициация подсказки
@SuppressLint("ResourceAsColor")
private void InitCheckCast(HitTestResult hit) {

    if (mTextView != null) {
        //Удаляем прошлую подсказку
        mLayout.removeView(mTextView);
        mTextView = null;
    }

    mTargetPoint = hit.getPoint();
    mTargetNode = hit.getNode();
    if (mTargetNode == null)
        return;

    boolean isInfoPanelActive =
mArAnimalCell.findViewById(R.id.layoutInfo).getVisibility() ==
View.VISIBLE;
    if (mArAnimalCell.getAnimalCell().getIsWorking() &&
!isInfoPanelActive) {
        mTextView = new
TextView(mArAnimalCell.getApplicationContext());
        mTextView.setLayoutParams(mLayoutParams);
        mTextView.setBackgroundResource(R.color.white);
        Info info =
AnimalCellElements.FindPartInfo(mTargetNode.getName());
        if (info != null)
            mTextView.setText(info.partName);
        else
            mTextView.setText("Животная клетка");

        Vector3 screenPoint =
mCamera.worldToScreenPoint(mTargetPoint);
        mTextView.setX(screenPoint.x);
        mTextView.setY(screenPoint.y);
        mTextView.setPadding(paddingInPX, paddingInPX,
paddingInPX, paddingInPX);
        mLayout.addView(mTextView);

        mFrameCounter = 2;
    } else {
        Info info =
AnimalCellElements.FindPartInfo(mTargetNode.getName());
        if (info != null)
            mPanelController.Activate(info.partName,
info.partDescr);
        else
            mPanelController.Activate("Деталь двигателя", "Деталь
конструкции двигателя");
    }
}
}

```

Приложение В

РЕЗУЛЬТАТЫ СТРУКТУРНОГО ТЕСТИРОВАНИЯ

(рекомендуемое)

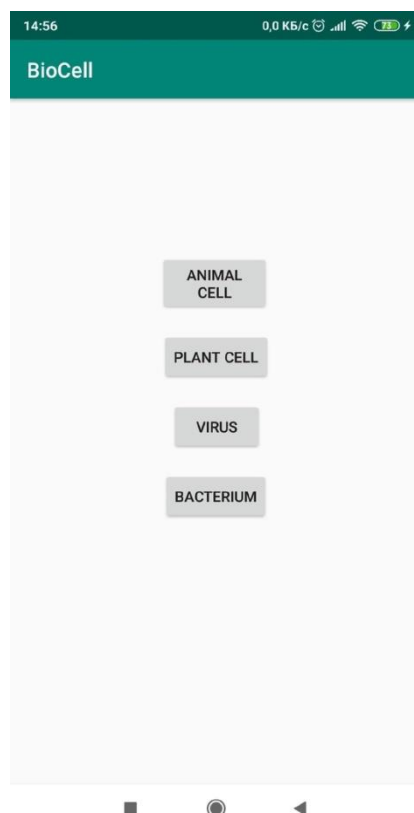


Рисунок В.1 – Пользователь не нажимал на кнопку выбора модели



Рисунок В.2 – Телефон пользователя не поддерживает AR

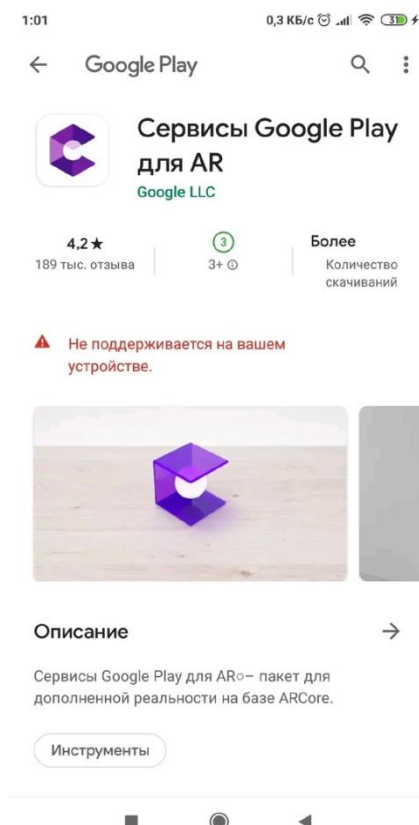


Рисунок В.3 – Сообщение пользователю о том, что его устройство не поддерживает режим AR

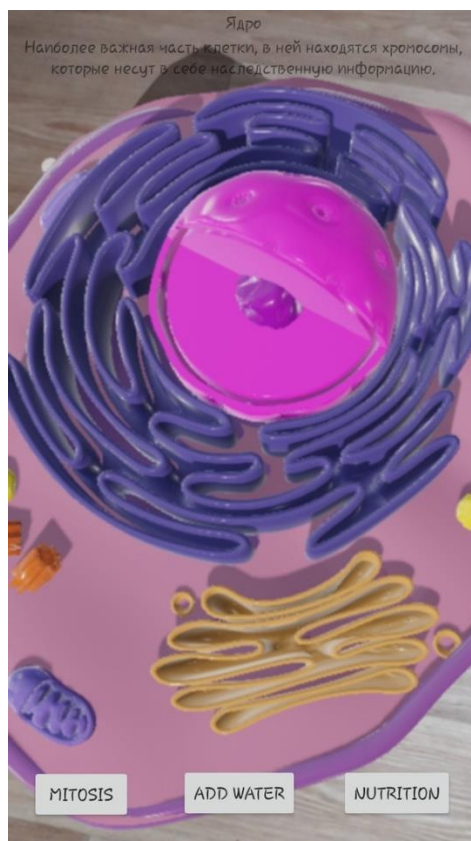


Рисунок В.4 – Модель клетки загружена

Приложение Г

АКТ О РЕЗУЛЬТАТАХ ВНЕДРЕНИЯ
(рекомендуемое)

ООО «СЕТАП»site: setupit.org

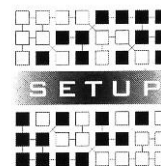
ОГРН: 1177746688416

ИНН: 7704434695

email: info@setupit.org

119435, г. Москва, ул. Пироговская М.,

д.6/4, корп. 2, пом. V КОМ 5

**Акт**

о внедрении результатов
выпускной квалификационной работы
Ефакиной Евгении Сергеевны

Подсистема приложения дополненной реальности «Живая клетка» по курсу биологии, разработанная стажером ООО «СЕТАП» Ефакиной Е.С. в ходе выпускной квалификационной работы, включена в программный комплекс Experiment и является его неотъемлемой частью. Данное решение разрабатывается в рамках образовательной онлайн-платформы Experiment.

Генеральный директор ООО «СЕТАП»

Хованский Д.Н.

26.06.20