Федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технологический университет»

(ФГБОУ ВО «КНИТУ»)

КУРСОВОЙ ПРОЕКТ

по дисциплине «Методы и средства проектирования информационных систем и технологий»

Тема: Обучение на основе виртуальной реальности сборке разборке изделий машиностроения

**Выполнил:**

студент группы 7201-11

Туйчиев Равшан Рустамович

**Проверил:**

зав. каф. ИПМ, доцент

Старыгина С.Д.

**Оценка:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Казань, 2023 г

Содержание

Введение…………………………………………………………….....3-4

1. Анализ предметной области, цели и постановка задачи……………..5
   1. Формирование общих требований……………………………...6
2. Фаза исследования и уточнения…………………………………….....7
   1. Концептуальная модель на DFD…………………………...…8-9
   2. Декомпозиция концептуальной модели на IDEF-3……..….9-14
   3. Рабочая (контекстная) модель …………………………….…..15
3. Выбор средств разработки………………………………………........16
4. Архитектура ПО………………………………………………...…17-18
5. Требования к программному продукту ………………………….19-20
6. Фаза тестирования…………………………………………………21-23

Заключение………………………………………………………….…24

Список литературы……………………………………………………25

Приложения (программный код)……………………………...….26-34

**Введение**

Информационные технологии не стоят на месте и стремительными темпами охватывают все новые и новые сферы деятельности человека. Уровень и доступность технологий достигли такого масштаба, что даже ученики средней школы, дома за самым простым компьютером имеют возможность создавать целые миры в виртуальной среде, достаточно иметь желание и упорство, чтобы узнавать новое и творить.

Так некоторые разработчики создают игры, в которые играют подчас миллионы людей. А другие, что более нам интересное, создают на основе тех же самых игровых движков прикладные инструменты. Взять тот же самый Car Mechanic Simulator разработанный компанией Red Dot Games. Хоть формально и являющийся игрой, по своей сути получается прикладным инструментом. В нем вам дается возможность почувствовать себя на месте автомеханика, производить те же действия, что и реальные механики на станциях технического обслуживания автомобилей. Производить те же операции. И что самым интересным и в то же время полезным открывается возможность перенести этот опыт на другие реальные практики человеческой деятельности.

Нынешние технологии дают возможность не только создавать простые симулятор популярных активностей, будь то гонки, полеты на самолетах и т.д., но к тому же делать их индивидуальными, например, добавив определенные механизмы в приложения и разработав их логику можно спокойно и в огромных объемах с минимальными затратами ресурсов обучать новый персонал работе с подчас опасным оборудованием.

Не смотря на все плюсы симуляторов, процесс их создания имеет невообразимое количество человеческих ресурсов, и порой требует даже внушительную команду разработчиков, поэтому следует проанализировать аналоги создаваемого продукта. Как минимум для того, чтобы достичь тех же результатов, что и они и как максимум не допустить тех же ошибок, что были ими допущены. Перед созданием, нельзя обойти стороной определение архитектуры будущего продукта и его внешнего вида. Нужно четко понимать, кто будет пользоваться данным продуктом, какая возрастная категория, на каких платформах будет данный продукт выпускаться и еще много чего.

Работу по созданию программного продукта, который по определению является информационной системой, как и любого другого проекта, начинается с того, что нужно четко и ясно, представлять и учитывать требования, выдвинутые заказчиком к конечному продукту в виде документа ТЗ. Техническое задание ТЗ – это документ, или их совокупность, описывающие цель, структуры и свойства проекта, которые должны быть воплощены.

1. **Анализ предметной области, цели и постановка задачи**

Целью данной работы является создание программного продукта, позволяющий в виртуальной среде взаимодействовать с объектами машиностроения. Заказчиками и в том числе конечными пользователями реализуемого программного обеспечения являются производственный предприятия и обучающие организации. Деятельность таких компаний должна быть связана с производством, конструированием, сборкой или обслуживанием сложного оборудования, требующая высокую квалификацию для данных типов работ.

Обучающий процесс в подобного рода компаниях, согласно полученными мной информацией, организован таким образом, что обучающиеся проектируют, конструируют, агрегируют и разбирают сложное оборудование. Данные процессы, в ходе своего выполнения влекут за собой издержки: затраты физических ресурсов, огромные затраты времени, а также, что очень важно и нельзя оставить без внимания риск для здоровья.

Издержки физических ресурсов заключаются в том, что при вышеописанных процессах, все действо происходит с реальными физическими объектами, на создание которых требуются много сил, времени и природных ресурсов. Так если какой-то студент, при не удачной сборке каким-либо образом повредит деталь, то дальнейшее её использование будет маловероятно. При этом во время обучения данное событие априори неизбежно. В конечно итоге это выливается в огромные счета для компании, требующие оплату за поврежденный инвентарь и детали.

Затраты времени, на первый взгляд, не такие очевидные, как может показаться. Однако целевая деятельность требует, из-за фактора обучения, долгосрочное, с точки зрения сессии многочасового вовлечения студента, внимание к конкретному процессу. Можно заметить, что для ускорения обучения, компании проводят курсы, занятия в группах, что требует отдельных затрат на организацию и подготовку конкретного экземпляра продукта к работе.

Риск для здоровья уже более очевидный и понятный фактор издержек. В данной области он проявляет себя в возможностях обучающимися получить травмы во время сборки, либо работы по обслуживанию конкретных изделий машиностроения. Так, например, поставив деталь в неправильную позицию, при запуске оборудования может произойти, либо разлет составных частей и их деформация. Что приводит как минимум с лишним затрат ресурсов, а максимум может повлечь тяжкий вред жизни человека, если даже не летальны исход.

Исходя из вышеописанных издержек обучения, можно выделить бросающиеся в глаза проблемы компаний, требующие решения.

Проблемы:

* Снизить издержки процесса обучения
* Повысить безопасность целевых процессов
* Повысить эффективность обучения

Проблема повышение эффективности обучения мною проглядывается в том, что единоличный процесс обучения по сравнению с групповым является более эффективным. Сразу стоит отметить, что этот тезис не применим ко всему процессу обучения, а лишь только к конкретному. А именно ситуация, при которой один обучающийся работает с оборудованием, в то время как другие лишь тихо и смирно стоят в стороне и наблюдают, по понятной причине – оборудование в одном экземпляре, из-за своей дороговизны.

Возвращаясь к программному продукту, можно увидеть в нем инструмент для решения приведенных проблем. Начиная с наиболее важного – безопасность жизни и деятельности. Виртуальная среда имеет максимальный уровень безопасности, беря во внимание незначительную нагрузку на зрительную систему и другие последствия, сопровождаемые работой с электронными устройствами, работа производится с цифровым двойником, а не физическим оригиналом и чисто логически понятно, что никакой физический тяжкий вред в таких условиях не может быть получен.

Передвигаясь от более значительных к менее значительным проблемам, рассматриваем решение снижения издержек. Виртуальная среда, не требует огромных физических ресурсов, для представления свойств реальных объектов т.е. не требуется физический экземпляр, имеющий ощутимую стоимость. Цифровой двойник не может быть поврежден, по очевидным причинам, что снижает издержки ресурсов. Не придется закупать множество деталей и оборудований по несколько раз, потому что единожды созданный качественный цифровой двойник, будет передавать основные свойства реального объекта и давать бесконечную возможность на создании копий и многократного и одновременного использования.

Одновременное использования решает следующую проблему: повышения эффективности обучения. Стажеры и студенты смогут, благодаря виртуальной среде, параллельно производить манипуляции каждый со своим цифровым двойником, а не быть зрителем в стороне. Ведь один из важнейших принципов в обучении это практика. Так каждый обучающийся, своими руками, вместе со всеми будет выполнять целевые задачи, а не ожидать в очереди к дорогостоящему оборудованию.

Имея достаточно полно описанные проблемы, можно поставить задачи проекта.

Задачи:

* Моделирование архитектуры программного обеспечения
* Разработка MVP ПО удовлетворяющего нашим ожиданиям
* Тестирование и пилотный ввод в эксплуатацию

MVP – минимально жизнеспособный продукт (eng. Minimum Viable product), продукт обладающий минимальными, но достаточными для удовлетворения первых потребителей функциями. Основной задачей которого является сбор обратной связи для дальнейшего формирования гипотез и уточнений требований и задач дальнейшего развития продукта.

* 1. **Формирование требований**

Требования к программному продукту.

Поскольку любой программный продукт является системой, а подчас и даже сложной суперсистемой, то рассматривать его и описывать требования нужно с точки зрения систем. А наименьшей, функциональной, неделимой частью системы является элемент. Таким образом требования к системе представлены в виде элементов системы, которые должны быть в нее включены.

Требования, элементы системы.

* Передвижение пользователя в пространстве
* Цифровой двойник реального объекта
* Взаимодействие пользователя и цифровых двойников
* Чертеж собираемой модели
* Логика отображения чертежей
* Создание цифровых двойников в виртуальной среде
* Окружающее виртуальное пространство

1. **Фаза исследования и уточнения**

В фазе исследования будут представлены:

* IDEF3 модель (каждый блок по отдельности, для более подробного и точного объяснения)
* Описание элементов системы и их свойств
  1. **IDEF3 модель и разбор на составляющие**

IDEF3 нотация, она же методология моделирования и общепризнанный стандарт предназначенный для документирования процессов, происходящих в системе. Данная нотация описывает связи – причины и следствия, между ситуациями и событиями, являющиеся подсистемами, в доступном для профессионалов представлении.

Описание этапов в IDEF3 происходит графическим образом. Показываются взаимосвязи функций и объектов друг с другом, а также обработка информации. Данный стандарт отличный инструмент для аналитики, поскольку дает возможность детально описать отдельные этапы или сущности, которые принимают участие в одном и том же процессе.

* 1. **Декомпозиция модели проектирования на IDEF-3**

Каждый отдельный этап в IDEF-3 модели содержит задания, отношения и пересечения.

Первый блок включает в себя несколько задач, которые требуется решить, прежде чем приступать к следующем этапу, со схемой можно ознакомиться на рис.1. Ниже приведен список задач данного этапа с описанием блоков, идущих по порядку, согласно логики процесса разработки ПО.

Получение задания – начальный этап при любом проектировании, без которого чисто физически и логически не будет возможности для дальнейших действий. На данном этапе узнаётся желаем результат конечного продукта, какие свойства он должен иметь, какие технологии и ресурсы должны быть использованы в ходе реализации проекта. Так же формулируются проблемы, которые должен решать конечный продукт. На основании проблем формируются задачи, которые должны быть решены с соблюдением заявленных требований.

Описание функционала – прежде чем приступать к самому процессу разработки: написание программного кода, создание макетов и т.д., требуется четко понимать, какие возможности нужно будет реализовывать. Потому что, когда задача описана по пунктам, с указанием свойств и желаемого результата, дальнейшая разработка становится многократно качественнее и быстрее. Это обусловлено физиологией мозга человека, который способен долго удерживать фокус внимания лишь на конечном, представимом списке образов и задач, требующих решения

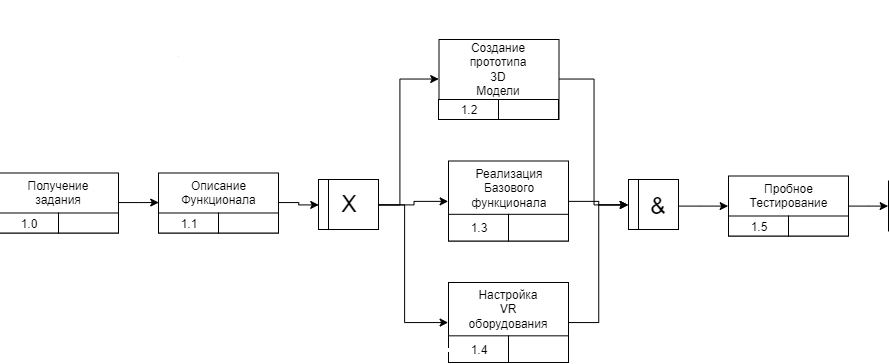
Следующие этапы происходят параллельно, т.е. можно приступить к любому из них или начать сразу все одновременно, поскольку это дает гибкость при работе. Так как этапы не сильно связаны друг с другом, то можно использовать максимум возможностей от каждого момента времени разработки.

Создание прототипа 3D модели – на данном этапе выбирается первая модель, которая будет цифровым двойником на этапе проектирования. Для повышения качества и эффективности конечного продукта, на данном этапе эта модель должна быть среднего уровня детализованности. То есть состоять не из малого количества составных деталей будь то три или четыре части и не из огромного множества, на создание и настройку которых уйдет несколько месяцев работы. Дюжина деталей является предельно достаточным для данного этапа. Поскольку этого количества хватит для описания функционала, который впоследствии может быть экстраполирован на другие аналогичные компоненты.

Настройка VR оборудования – необходимый этап, реализация которого дает возможность разработки компонентов, предназначенных для интегрирования VR контроллеров со средой разработки. В дальнейшем это даст возможность, производить все действия в виртуальной среде при помощи обеих рук и головы используя все 3 степени свободы пространства. Что по сравнению с обычными клавиатурой и мышью является более эффективным и качественным вариантом.

К следующему этапу можно приступить лишь только при выполнение всех трех предшествующих этапов разработки, поскольку интегрирование результатов этих трех этапов, даст более полное начальное представление о конечном продукте.

Пробное тестирование – этап без которого невозможно проводить разработку программного продукта. На данном этапе мы пытаемся «сломать» приложение, используя лишь те возможности, которые были даны пользователю. Таким образом находятся ошибки, которые мог встретить пользователь во время реальной работы с программой и получить негативный опыт, вследствие которого мог навсегда отказаться от использования данного приложения.



**Рис. 1. Первый блок**

* Второй блок – ознакомление с макетом сайта. Здесь нужно учитывать, что оба требования обязательны для выполнения. Просмотр макета – это ознакомление с внешним видом, как будет показан итоговый продукт. Работа этого блока продемонстрирована на рис. 2

Ознакомление с требованиями – учет современных требований для создания, так называемый Code Style.



**Рис. 2. Второй блок**

Второй блок – это следующий цикл по спиральной модели разработки, при котором уточняются, расширяются возможности, свойства и функции описанные в предыдущем цикле разработки. Также, как и на предыдущем этапе важно учитывать каждое требование. Работа этого блока продемонстрирована на рис. 2.

На данном этапе имеется возможность осуществлять разработку сразу по нескольким направлениям, поскольку предыдущие этапу выполнены и проект позволяет. Сам действия могут быть делегированы среди разных специалистов, это отличное подспорье для дальнейшего коммерческого развития продукта, поскольку командная работа дает огромные бонусы при работе над проект, учитывая, что модель проект позволяет параллельное исполнение множества задач.

Важное требование, которое должно соблюдаться на протяжении всего проекта это стиль кода, он же чистый код. Поскольку C# язык используется для описание сложных функций, то чистый код - это необходимый компонент, который позволит в дальнейшем поддерживать разработанный продукт. Ибо, написав нечто понятное сейчас, при взгляде на тот же самой программный код спустя даже месяц, может потребоваться колоссальное количество времени чтобы просто вникнуть в суть происходящего, не говоря уже об изменении функционала.

Создание детального прототипа – на этом этапе предыдущий базовый прототип дорабатывается, уточняются детали конкретного объекта. Если до этого прототип первого цикла, представлял собой простые фигуры будь то куб, сфера или цилиндр, то на данном этапе геометрия деталей и конечного объекта в целом становится узнаваемой. Так узкий цилиндр, приобретает форму колеса с резиной и ободом. Взглянув на который, любой дееспособный человек ответственно сможет заявить, что это колесо. И так абсолютно со всеми деталями. Так же на этой фазе создаются материалы, которые будут покрывать поверхность каждой детали, что будет повышать эффективность интуитивного определения сущности каждого конкретного объекта.

Уточнение и расширение функционала – если до этого у нас были базовые функции, дающие возможность перемещаться в пространстве, вращать камерой, и передвигать объекты, а сам чертеж лишь статично без всякой реакции на действия пользователя находился в пространстве. То теперь на этом этапе добавиться реакции чертежа на действия пользователя, так что он сможет при перемещении детали в пространстве той же детали на чертеже установить её и это будет сопровождаться визуальными изменениями.

Обработка ошибок – безусловный компонент, который невозможно пропустить. На предыдущем этапе было обнаружено множество ошибок, так называемых багов. Когда ожидалось одно поведение от сущности, а по факту происходит совершенно другое. Одной из краеугольных ошибок, которое могло поставить под вопрос возможность реализации проекта была оптимизация. На одной из стадий разработки FPS(Frames Per Second eng. Кадров в секунду. Общепризнанный параметр, указывающий на эффективность использования ресурсов компьютера приложением) в демонстрационном режим падал до 10-20 кадров в секунду, что является недопустимым. Множество возможных причин появления этой и проблемы и пути их возможных решений были обработаны, что внесло существенную долю издержек времени.

Только когда все эти три этапа выполнены, мы можем перейти к предрелизному тестированию, на котором происходит так называемый полишинг (polishing eng. Полировка, общепринятый в среде разработчиков компьютерных игр термин, обозначающий процесс доработки мелких деталей, которые на первый взгляд не бросаются в глаза)

Этап презентации содержит в себе само создание презентации готового продукта. Описания достигнутого результата в ходе его реализации и так же сама демонстрация перед людьми.

1. **Выбор средств разработки**

Код используемый в проекте был написан в IDE Visual Studio 2022. IDE -интегрированная среда разработки (integrated development enviroment), программный комплекс включающий в себя: текстовый редактор, компилятор, средства отладки. Все это способствует ускорению процесса разработки. Visual Studio 2022 – это бесплатный редактор кода, который поддерживает множество языков программирования, в том числе и объектно-ориентированный язык программирования C#, разработанный компанией Microsoft. Этот редактор без сомнения удобен тем, что имеет объединение с Git, автоматическим дополнением разрабатываемого кода, подсветке синтаксиса, возможности отладки и быстрому доступу к файлам проекта. Помимо этого, редактор имеет удобную интеграцию с другим средством разработки – Unity. Постоянные обновления делает его, по моему мнению, одним из лучших редакторов для разработки приложений на языке C#.

C# - Высокоуровневый объектно-ориентированный язык программирования, поддерживающий в себе все принципы ООП, такие как полиморфизм, наследование, перегрузка операторов, статическую типизацию. На этом языке программирования разработано множество программ на самые разные платформы начиная с приложения на мобильные устройства и заканчивая веб-приложениями. Наиболее часто на C# пишут программы для десктопных устройств, игры и веб-сервисы.

Unity – межплатформенная среда разработки компьютерных игр. Межплатформенность дает возможность разрабатывать один продукт, выпуская его на множество платформ, начиная с наиболее популярных Windows, macOS, Android, iOS и заканчивая портативными консолями и веб-приложениями, что делает его объединяющим (Unity с англ. - единство). Самым большим достоинством, которыми может похвастаться Unity это современный графический интерфейс с Drag&Drop интерфейсом, позволяющий перетаскивать с помощью мыши множество объектов разработки (звуки, 2D и 3D объекты, программный код, модули). Unity Asset Store – популярный магазин содержащий готовые решения для множества разрабатываемых проектов, отличное подспорье, ускоряющий процесс разработки в разы.

КОМПАС-3D – Российский программный САПР комплекс, позволяющий оперативно создавать и выпускать чертежи сложных изделий, таблицы, макеты и прочие документы. САПР технология – система автоматизированного проектирования, позволяющий эффективно с минимальными затратами времени и ресурсов оперативно проводить разработку моделей, макетов и чертежей. Данный инструмент стал стандартом при проектировании среди множества производственных компаний.

Blender – Программное обеспечение используемое при разработке 3D моделей. Имеет плотную интеграцию со средой разработки Unity, что делает его отличным кандидатом. Данный инструмент распространяется в том числе и по свободный лицензии, что делает его доступным и бесплатным.

1. **Архитектура ПО**

При разработке любой информационный системы к ней выдвигаются требования. Требования представляют из себя список достаточно полном описывающие границы и свойства, которые обязательно должны соблюдаться. Так и архитектура приложения имеет ряд требований, которые должны быть исполнены и соблюдены.

Одними из основных требований к архитектуре ПО являются такие компоненты как: надежность, масштабируемость, быстродействие и безопасность.

В настоящее время наиболее распространенными типами архитектурами программного обеспечения являются: Многослойная, Многоуровневая, Сервис-ориентированная, Микросервисная.

Поскольку наш программный продукт по сути своей будет является неким симулятором, наибольшая часть вычислений происходит на самом клиенте т.е. компьютере пользователя, то представляется рациональным использовать любую другую архитектуру, кроме как многослойной. Остальные три архитектуры подразумевают под собой очень частый обмен информацией между разными модулями нескольких клиентов пользователей, чего в нашем продукте не предполагается. Подключение, да и создание серверной части просто не требуется, ввиду отсутствия задач, которые она могла бы решать в данном проекте. Отсюда многослойная архитектура представляется в виде лучшего кандидата, поскольку имеет такие преимущества:

* Более простая реализация по сравнению с другими подходами. То есть нам не потребуется огромную часть времени тратить лишь только на изучение какой-либо технологии, которая не будет решать никакой задачи либо это задача будет составлять менее процента от всего объема работ.
* Возможность абстракции, благодаря разделению обязанностей между уровнями. Возможности применения объектно-ориентированного подхода, что делает более простым процесс разработки, благодарю человеческому представлению сложных компонентов системы.
* Изолированность элементов, защищающая одни слои от вмешательства в них других слоев. Так логика отображения объектов не пересечется и не приведет к ошибкам, с логикой поведения самих объектов.
* За счет слабой связи компонентов, мы получаем хорошую управляемость программным продуктом во время разработки, а также возможность модифицируемости и расширения её в дальнейшем.

1. **Требования к программному продукту**

Требования к программному продукту неотъемлемая часть абсолютно любого проекта. Поскольку только благодаря требованиям можно понять какие основные части требуются для того чтобы реализовать выбранный проект. Требования бывают разные и каждое из них описывает разные включенные в проект сферы человеческой деятельности, несмотря на то что все это происходит в одном и том же проекте.

Прежде всего, следует отметить важность бизнес-требований, которые определяют цели заказчика и направлены на обеспечение удобства для пользователей или клиентов. Эти требования предполагают, что продукт будет полезным инструментом как для конечного пользователя, так и для заказчика.

Для пользователей это удобная возможность работы в самой программе, интуитивности при изучении функциональности. Для заказчика – понимание того, что данный продукт решает его задачи и это можно увидеть.

Пользовательские требования – эти требования дают образ, как должен выглядеть продукт со стороны пользователя. В этом проекте к пользовательским требований можно отнести: отображение деталей, имеющую значительно похожую форму, как и физический оригинал; корректность этапов сборки, при котором не будет возможности грубо говоря установить колесо на шасси которого еще нет.

Атрибуты качества – данные качества состоят из стандартов и теории по оценке и обеспечению качества.

В этом проекте к атрибутам качествам можно отнести быстродействие работы программы, при котором количество кадров в секунду не опускается нижу значения в 30 единиц. А также возможность запуска на разных платформах.

Нельзя забыть отметить тот факт, что описанные примеры не в полной мере являются атрибутами качества, а скорее требованиями к атрибутам. Быстрая работа приложения является производительностью, а процесс по решению проблемы производительности – оптимизацией.

Функциональные требования – это те требования, которые описывают, как должна функционировать система, и должны быть представлены в различных форматах. Они не входят в другие категории если объяснить просто.

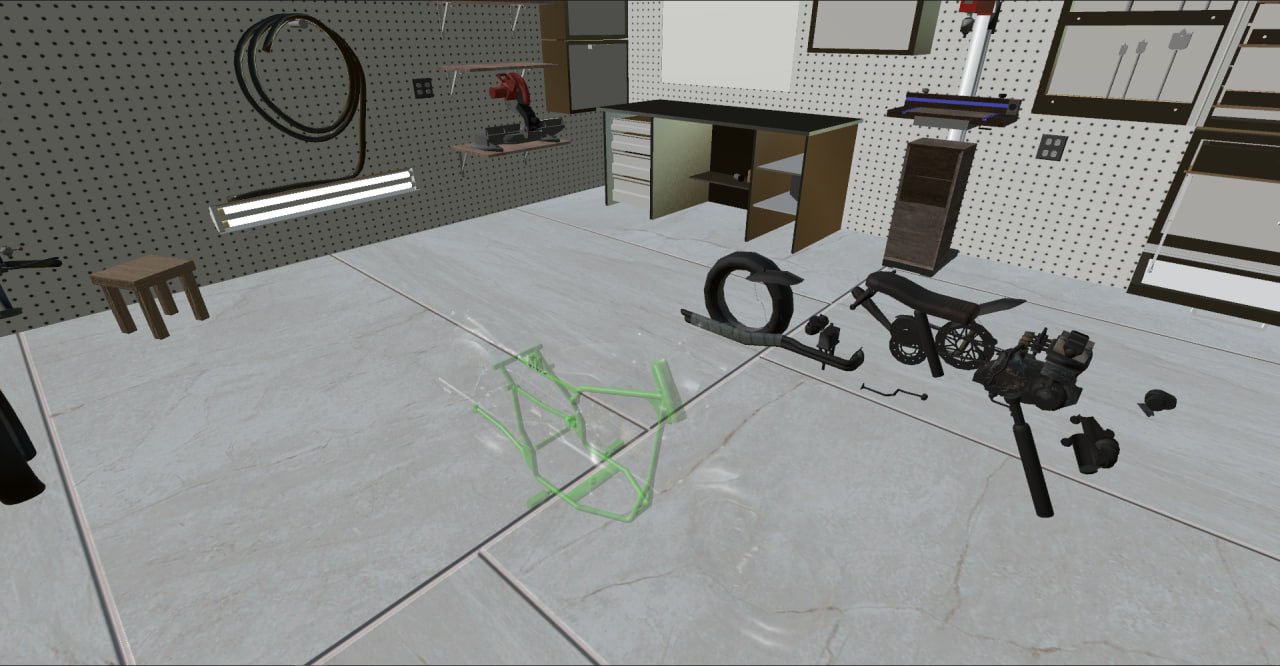
В данной работе к функциональным требованиям смело можно отнести перемещение в пространстве, вращение камерой, отображение деталей и чертежей и возможность сборки продукции.

1. **Фаза тестирования**

Тестирование — одна из важнейших частей процесса разработки любого программного продукта. Во время тестирования подтверждаются, либо опровергаются теории, которые были выдвинуты или реализованы на предыдущей стадии разработки. Происходит выявление ошибок, которые во время разработки не были обнаружены либо даже мысль о возможности ошибок такого рода даже не появлялись.

Видов тестирования огромное множество и в современных реалиях компании, работающие в IT сфере, даже имеют целые команды тестировщиков для данной работы. То есть эта часть работы настолько огромная, что разделение труда дошло до такого уровня. Ввиду огромного объема работы, в данном проекте было использовано лишь функциональное ручное тестирование. При котором во время работы программы, мной проверялись работоспособность функций реализованных на предыдущих этапах.

Поскольку печатный документ не дает возможности включения видеоматериалов процесса работы программы, то ниже будут приведены лишь только скриншоты демонстрационной версии, с описанием происходящего процесса и выполненной работы.



**Рис. 3. Начальное представление**

На рис.3 сразу можно увидеть множество объектов, бросающихся в глаза. Пойдем по порядку, от менее значимого, к более.

Во первых сразу видно окружение, которое представляет из себя гаражную мастерскую. Поскольку работу в пустом пространстве, где вас окружало бы лишь пустота, просто недопустимо. А воображение пользователя того, что он является неким мастером, дает огромное положительное психологическое влияние на процесс обучения.

Далее можно увидеть расположенные в случайном порядке детали собираемого объекта, в данном случае мотоцикла. При запуске программы, расположение деталей задается случайно, поскольку в противном случае, пользователь бы просто машинально тянулся к тому же самому месту, где всегда появляется конкретная деталь. Во время обучения это не допустимо, поскольку лишает возможности научиться человеку визуально определять детали среди таких же похожих.

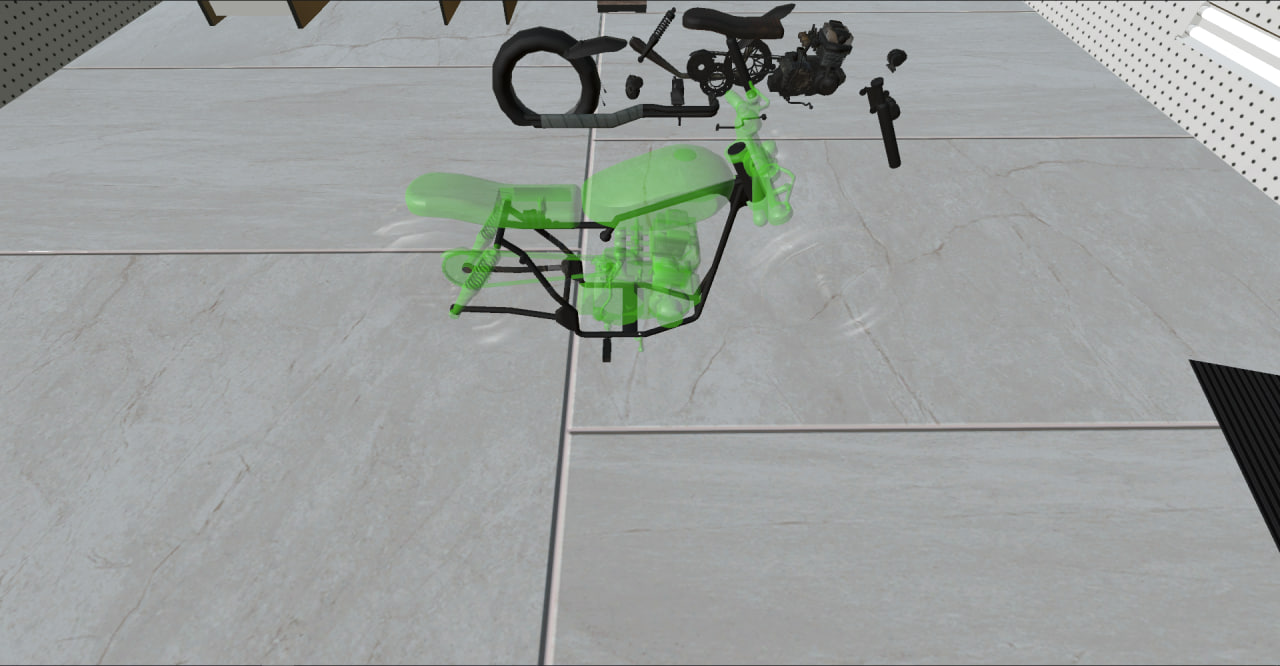
Далее видно основную несущую раму мотоцикла, она имеет бледно зеленый, прозрачный цвет, что дает интуитивное понимание, что это чертеж. И как выглядит деталь, которую требуется поставить в конкретное место.

Так же если внимательней приглядеться, можно различить почти невидимые остальные детали, собираемого объекта. Это дает представление о конечном результате, который может быть достигнут и то, что это результат очень близок.



**Рис. 4. Демонстрация перемещния**

На следующем рис.4 демонстрируется возможность перемещения пользователя в пространстве, возможность вращения камерой. А также, что очень важно, возможность «брать в руки» и перемещать объекты в пространстве, а так же ставить их в желаемом месте.



**Рис. 5. Поэтапная сборка**

С реализацией функциональности, которую видно на рис. 5, можно будет ознакомиться в приложении к данной работе.

На данном рисунке отчетливо и интуитивно понятно, что первая деталь уже установлена, и пользователю дается возможность и визуальная подсказка, говорящие о том, какие на следующем этапе сборки, должны быть присоединены детали и в каком конкретном месте они должны находиться.

Весь этот процесс просчитывается динамически, так после установки следующей детали, будут новые возможности для установки.

Нельзя не отметить также тот факт, что после установки детали на место чертежа, она принимает визуальное отображение именно самой детали, а не чертежа, т.е. зеленый чертеж данной детали уже не видно.



**Рис. 6. Визуальная подсказка**

На следующем рис. 6 продемонстрирована другая функциональность повышающая уровень интуитивности во время сборки. Пользователь держит деталь, которая представляет из себя двигатель и в то же самое время, как только пользователь взял в руку деталь, чертеж меняет свой цвет на желтый, давая понять, в какое конкретное место должна быть установлена деталь.

**Заключение**

В результате реализации данного проекта, курсовой работы, было разработано программное обеспечение для одной из кафедр нашего вуза, занимающейся инженерной графикой. Были использованы такие инструменты как Unity, VS studio 2022, КОМПАС-3D и Blender. В ходе работ весь программный код был написан на популярном языке C#. Как результат работы стоит отметить не только созданный программный продукт, но и повышение навыков работы с вышеуказанными инструментами и что не мало важно для программисты более углубленное, практическое знание языка C#.

Эта работа состоит из анализа требований, постановки задач из выявленных проблем, самих этапов реализации проекта дающее возможности для обучения и имеющие огромные перспективы.

Результатом выполнения данного проекта является разработанный программный продукт, имеющий ряд функций и свойств таких как:

* Возможность перемещения пользователя в пространстве и вращение камеры
* Возможность перемещения виртуальных деталей и взаимодействий с ними
* Возможность сборки оборудования, механизмов.
* Интуитивность в работе с приложением.
* Поэтапное отображение стадий сборки.
* Визуальное понимание свойств собираемых деталей.

**Список литературы**

1. Джозеф Хокинг – «Unity в действии. Мультиплатформенная разработка на C#», Питер, 2019, 352стр.
2. «Полное руководство по языку программирования С# 11 и платформе .NET 7» <https://metanit.com/sharp/tutorial/>
3. Майк Гейг – «Разработка игр на Unity 2018 за 24 часа», Эксмо, 2020, 464 стр.
4. Феронне Хариссон – «Изучаем C# через разработку игр На Unity», Питер, 2022, 400 стр.
5. Васильев Алексей Николаевич - «Программирование на C# для начинающих. Основные сведения», Бомбора, 2023, 592 стр.
6. Прайс Марк «C# 10 и .NET 6. Современная кросс-платформенная разработка», Питер, 2023, 848 стр.
7. Гибсон Бонд Джереми – «Unity и C#. Геймдев от идеи до реализации», Питер, 2019, 928 стр.
8. Бхаргава Адитья – «Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих», Питер, 2022, 288 стр.
9. Шрейер Джейсон – «Кровь пот и пиксели», Бомбора, 2022, 368 стр.
10. Стивенс Род – «Алгоритмы. Теория и практическое применение», Эксмо, 2022, 544 стр.
11. Евдокимов П. В. – «C#. Практическое руководство», Наука и техника, 2022, 416 стр.
12. Албахари Джозеф – «C# 9.0. Справочник. Полное описание языка», Эксмо, 2021, 1056 стр.
13. Джуст Виссер – «Разработка обслуживаемых программ на языке C#», ДМК Пресс, 2017, 192 стр.
14. Мюллер Джон Поль, Семпф билл, Сфер Чак – «C# для чайников», Диалектика-Вильямс, 2019, 608 стр.
15. Мартин Роберт – «Чистый код: создание, анализ и рефакторинг. Библиотека программиста», Питер, 2022, 464 стр.