Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет

«Высшая школа экономики»

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

ОТЧЕТ

по преддипломной практике

Подготовка ВКР на тему

Клиент-серверное мобильное приложение для синтеза аранжировок музыкальных произведений

Выполнил студент

образовательной программы

09.03.04 «Программная инженерия»

группы БПИ207

\_\_\_\_\_\_\_\_ \_\_В.К. Мельник\_\_\_\_\_

(подпись) (инициалы, фамилия)

**Руководитель практики от НИУ ВШЭ**

Доцент технических наук, профессор департамента Программной инженерии НИУ ВШЭ, Александров Дмитрий Владимирович

(уч. степень, должность, ФИО)

\_\_\_23.04.2024\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (оценка) (подпись)

**Оценка комиссии**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (оценка) (подпись)

Москва – 2024

**АННОТАЦИЯ**

В данном документе приведён драфт выпускной квалификационной работы «Клиент-серверное мобильное приложение для синтеза аранжировок музыкальных произведений».

Настоящий документ содержит следующие разделы: «Введение», «Основание для разработки», «Обзор алгоритмов генерации музыки», «Требования к программе», «Требования к программным документам», «Технико-экономические показатели» и приложения [7].

В разделе «Введение» указано наименование и краткая характеристика области применения программы.

В разделе «Обзор алгоритмов генерации музыки» описаны основные виды алгоритмов, применяющихся в генерации музыки, а также описание алгоритма, использующееся в предлагаемой программе.

В разделе «Технические характеристики» содержатся следующие подразделы:

1. Требования к функциональным характеристикам
2. Описание алгоритма функционирования программы
3. Требования к интерфейсу
4. Требования к надёжности
5. Условия эксплуатации
6. Требования к составу и параметрам технических средств
7. Требования к маркировке и упаковке
8. Требования к транспортировке

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.101-77 Виды программ и программных документов [1];
2. ГОСТ 19.102-77 Стадии разработки [2];
3. ГОСТ 19.103-77 Обозначения программ и программных документов [3];
4. ГОСТ 19.104-78 Основные надписи [4];
5. ГОСТ 19.105-78 Общие требования к программным документам [5];
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению [7].

Изменения к данному Техническому заданию оформляются согласно   
ГОСТ 19.603-78 [12], ГОСТ 19.604-78 [13].

Также рекомендуется ознакомиться с терминологией, которая используется в данном документе, в приложении 1.

Оглавление

[1. ВВЕДЕНИЕ 6](#_Toc164795620)

[1.1. Наименование программы 6](#_Toc164795621)

[1.2. Краткая характеристика и область назначения 6](#_Toc164795622)

[2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ 7](#_Toc164795623)

[2.1. Документы, на основании которых ведется разработка 7](#_Toc164795624)

[3. НАЗНАЧЕНИЕ РАЗРАБОТКИ 8](#_Toc164795625)

[3.1. Функциональное назначение 8](#_Toc164795626)

[3.2. Эксплуатационное назначение 8](#_Toc164795627)

[3.3 Описание области применения 8](#_Toc164795628)

[4. ОБЗОР АЛГОРИТМОВ ГЕНЕРАЦИИ МУЗЫКИ 9](#_Toc164795629)

[4.1. Обзор видов алгоритмов генерации музыки 9](#_Toc164795630)

[4.2. Описание генетического алгоритма 9](#_Toc164795631)

[5. ТРЕБОВНИЯ К ПРОГРАММЕ 11](#_Toc164795632)

[5.1. Требования к функциональным характеристикам 11](#_Toc164795633)

[5.1.1. Требования к составу выполняемых функций. 11](#_Toc164795634)

[5.1.2. Организация входных данных 12](#_Toc164795635)

[5.1.3. Обоснование организации входных данных 12](#_Toc164795636)

[5.1.4. Организация выходных данных 13](#_Toc164795637)

[5.1.5. Обоснование организации выходных данных 13](#_Toc164795638)

[5.2. Описание алгоритмов функционирования программы 13](#_Toc164795639)

[5.3. Требования к интерфейсу 17](#_Toc164795640)

[5.4. Требования к надежности 22](#_Toc164795641)

[5.4.1. Требования к обеспечению надёжного (устойчивого) функционирования программы 22](#_Toc164795642)

[5.4.2. Время восстановления после отказа 23](#_Toc164795643)

[5.4.3. Отказы из-за некорректных действий оператора 23](#_Toc164795644)

[5.5. Условия эксплуатации 23](#_Toc164795645)

[5.5.1. Климатические условия эксплуатации 23](#_Toc164795646)

[5.5.2. Требования к видам обслуживания 23](#_Toc164795647)

[5.5.3. Требования к численности и квалификации персонала 23](#_Toc164795648)

[5.6. Требования к составу и параметрам технических средств 23](#_Toc164795649)

[5.7. Требования к информационной и программной совместимости 23](#_Toc164795650)

[5.7.1. Требования к информационным структурам и методам решения. 23](#_Toc164795651)

[5.7.2. Требования к исходным кодам и языкам программирования 24](#_Toc164795652)

[5.7.3. Требования к программным средствам, используемым программой 24](#_Toc164795653)

[5.8. Требования к маркировке и упаковке 24](#_Toc164795654)

[5.9. Требования к транспортировке и хранению 24](#_Toc164795655)

[6. ТРЕБОВНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ 25](#_Toc164795656)

[6.5. Состав программной документации 25](#_Toc164795657)

[6.2. Специальные требования к программной документации 25](#_Toc164795658)

[7. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 26](#_Toc164795659)

[7.2. Ориентировочная экономическая эффективность 26](#_Toc164795660)

[7.3. Предполагаемая потребность 26](#_Toc164795661)

[7.4. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 27](#_Toc164795662)

[8. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ 27](#_Toc164795663)

[8.1 Виды испытаний 27](#_Toc164795664)

[8.2 Общие требования к приемке работ 28](#_Toc164795665)

[ПРИЛОЖЕНИЕ 1 29](#_Toc164795666)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 30](#_Toc164795667)

# ВВЕДЕНИЕ

# Наименование программы

Наименование программы – «Клиент-серверное мобильное приложение для синтеза аранжировок музыкальных произведений».

Наименование программы на английском языке – «Client-Server Mobile Application for Synthesizing Arrangements for Musical Compositions».

# Краткая характеристика и область назначения

Данная программа предназначена для помощи в написании музыкальных композиций.

Программа предлагает музыканту варианты аранжировки и развития предоставленной им мелодии. Функционал этой программы позволить пользователю автоматически синтезировать аккорды, подходящие к заданной мелодии, вариации этой мелодии, переложить её на другой лад и так далее. Таким образом данная программа может ускорить процесс написания композиции, предложить пользователю идеи для её улучшения и найти некоторые ошибки.

# ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

# Документы, на основании которых ведется разработка

Разработка ведется на основании утверждённой̆ академическим руководителем темы выпускной квалификационной работы в соответствии с учебным планом подготовки бакалавров (НИУ ВШЭ, факультет компьютерных наук) по направлению 09.03.04 «Программная инженерия».

Документ, на основании которого ведется разработка: Приказ Национального исследовательского университета "Высшая школа экономики" “Об изменении тем выпускных квалификационных работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук” от XX.XX.XXXX №XXXX.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

# Функциональное назначение

Программа состоит из двух частей – серверной части и мобильного приложения для платформы IOS. Серверная часть осуществляет синтез наборов нот на основе данной мелодии и различных параметров. Клиентская часть позволяет пользователю взаимодействовать с сервером и вводить данные с помощью микрофона и пользовательского графического интерфейса.

# Эксплуатационное назначение

Эксплуатационным назначением данной программы является синтез различных компонентов аранжировки музыкальной композиции на основе предоставленной пользователем мелодии. Программа синтезирует подходящую аккордовую последовательность, вариации мелодии, в том числе ритмические, синтезирует партию барабанов и других инструментов и так далее.

# Описание области применения

Клиент-серверное мобильное приложение предоставляет пользователю инструменты для генерации различных частей аранжировки данной пользователем мелодии, которые позволяют упростить процесс написания музыкальной композиции.

# ОБЗОР АЛГОРИТМОВ ГЕНЕРАЦИИ МУЗЫКИ

# Обзор видов алгоритмов генерации музыки

Развитие цифровых технологий изменило облик автоматического создания музыки с момента его появления на заре цифровых технологий. Несмотря на многочисленные достижения, такие вопросы, как музыкальные задачи, решаемые различными машинами, и степень их успеха остаются открытыми. [28]

Идея автоматической генерации музыки появилась задолго до появления компьютеров. Например, один из таких алгоритмов, авторство которого приписывают Моцарту, был опубликован в 1792 году. [29]

Первые автоматически сгенерированные композиции появились в середине 1950-х годов, примерно в то же время, когда появился концепт искусственного интеллекта.

Основные методы генерации музыки делятся на алгоритмы, основанные на правилах, эволюционные алгоритмы, алгоритмы машинного обучения и гибридные алгоритмы.

Алгоритмы, основанные на правилах, используют заданный набор правил, которые описывают тот или иной жанр.

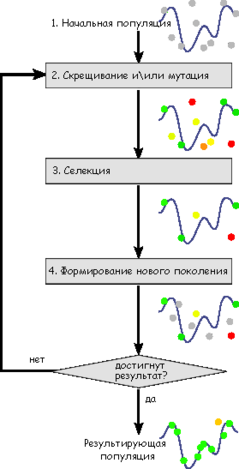
Эволюционные алгоритмы основаны на изменении популяции с помощью циклов отбора и изменения вариаций индивидов. Процесс начинается с инициализации популяции кандидатов решений, затем, в каждом цикле, некоторые из решений мутируют и рекомбинируют, пока не будет выполнено условие остановки алгоритма.

Алгоритмы машинного обучения добывают информацию из выборки. Пример алгоритмов машинного обучения – статистические модели, где свойства алгоритма определяются для того, чтобы назначить вероятности для автоматической генерации. К примеру, модели Маркова использовались для генерации мелодий. [30][28]

В клиент-серверном приложении, описанном в данном документе, предлагается использовать генетический алгоритм, в котором для отбора будут использоваться функции, основанные как на правилах, так и на статистических данных.

# Описание генетического алгоритма

1. Решение задачи кодируется с помощью вектора генов, где каждый ген может быть каким-то объектом.
   1. Решение может быть оценено с помощью «функции приспособленности»
2. Создаётся начальная популяция геномов, которые как правило инициализируются случайным образом.
3. Выбираются решения, к которым будут применяться генетические операторы – скрещивание и мутация. Например, могут быть выбрано определённое количество лучших решений.
   1. Алгоритмы скрещивания и мутации определяются, исходя из данных задачи.
4. При выполнении условия остановки алгоритма выбирается лучшее решение из текущей популяции. Также некоторые модификации алгоритма поддерживают лучшее решении на всех его итерациях для того, чтобы не потерять его на стадии мутации. [31]



(Рис. 1 – Генетический алгоритм)

# ТРЕБОВНИЯ К ПРОГРАММЕ

# Требования к функциональным характеристикам

# Требования к составу выполняемых функций.

Серверная часть:

1. Программа должна предоставить возможность зарегистрироваться или войти в аккаунт по почте и паролю
2. Программа должна предоставить возможность синтезировать последовательность аккордов подходящих по данной мелодии и параметрам, таким как:
   1. Количество аккордов
   2. Лад
   3. Музыкальный стиль
3. Предоставить возможность находить ошибки в предоставленной мелодии, таких как:
   1. Соответствие мелодии выбранному ладу и тональности
   2. Разрешение мелодии
4. Программа должна предоставить возможность транспонировать мелодию в выбранный лад
5. Программа должна предоставить возможность синтезировать вариацию мелодии в выбранном стиле с помощью:
   1. Перестановки частей мелодии
   2. Добавления нот
   3. Удаления нот
   4. Изменения длительностей нот
6. Программа должна предоставить возможность синтезировать барабанную партию к выбранной мелодии в выбранном стиле
7. Программа должна предоставить возможность синтезировать мелодию, хорошо звучащую одновременно с данной мелодией, в выбранном стиле
8. Программа должна предоставить возможность внести небольшие искажения в данный набор нот
9. Программа должна предоставить возможность сохранить созданную композицию
10. Программа должна предоставить возможность получить одну из сохранённых композиций

Клиентская часть:

1. Программа должна предоставить возможность зарегистрироваться или войти в аккаунт по почте и паролю
2. Программа должна предоставить возможность создать новый проект или открыть старый
3. Программа должна предоставить возможность изменить название проекта
4. Программа должна предоставить возможность ввести мелодию с помощью микрофона
   1. Программа должна распознать высоту сыгранных или пропетых пользователем нот
   2. Программа должна распознать длительности нот
5. Программа должна предоставить возможность вывести ноты в проекте на экран
6. Программа должна предоставить возможность удалить ноту
7. Программа должна предоставить возможность добавить ноту
8. Программа должна предоставить возможность скопировать часть нот
9. Программа должна предоставить возможность применить один из реализованных на сервере алгоритмов синтеза аранжировки и выбрать
   1. Музыкальный инструмент, который будет играть синтезированные ноты
   2. Музыкальный стиль
   3. Другие параметры, в зависимости от алгоритма
10. Программа должна предоставить возможность воспроизвести композицию виртуальными инструментами
11. Программа должна предоставить возможность экспортировать композицию в файл в формате MIDI
12. Проект должен автоматически сохраняться

# Организация входных данных

Входные данные для серверной части поступают через HTTP протокол в формате MIDI. Структура входных запросов должна соответствовать REST [15] спецификации.

Входные данные для клиентской части поступают с микрофона, встроенного в смартфон и через графический пользовательский интерфейс.

# Обоснование организации входных данных

Использование протокола HTTP является общепринятой практикой при создании клиент-серверных приложений.

Формат MIDI является общепринятым форматом записи музыки для работы в различном программном обеспечении.

Графический интерфейс позволяет точно ввести ноты в редактор. Ввод нот с микрофона позволит ускорить ввод данных.

# Организация выходных данных

Выходные данные серверной части отправляются через HTTP протокол в MIDI формате в iOS приложение. Структура входных запросов должна соответствовать REST [15] спецификации.

Выходные данные клиентской части показываются на экране смартфона с помощью графического интерфейса, выводятся с помощью встроенного в смартфон динамика или в виде файла в формате MIDI.

# Обоснование организации выходных данных

Использование протокола HTTP является общепринятой практикой при создании клиент-серверных приложений.

Формат MIDI является общепринятым форматом записи музыки для работы в различном программном обеспечении

Графический интерфейс позволяет наглядно представить информацию на экране, динамик позволяет пользователю лучше оценить полученные части композиции.

# Описание алгоритмов функционирования программы

* + 1. Алгоритм авторизации пользователя

1. На экране входа/регистрации пользователь заполняет поле электронной почты и пароля.
2. Пользователь нажимает кнопку «войти»
   1. Если пользователь найден в базе данных, происходит вход и навигация на следующий экран
3. Пользователь нажимает кнопку «зарегистрироваться»
   1. Если пользователя нет в базе данных, создаётся аккаунт и происходит навигация на следующий экран.
      1. Алгоритм взаимодействия клиента и сервера

Клиент-серверное взаимодействие осуществляется при помощи протокола HTTP. Ноты кодируются в формат MIDI и передаются в сыром виде в теле запроса. Дополнительная информация передаётся в параметрах запроса.

* + 1. Алгоритм редактирования проекта

Архитектура интерфейса редактора основана на архитектуре Model View Controller.

За отображение и обработку нажатий отвечает класс PianoRollView, который является наследником View из фреймворка SwiftUI. PianoRollView представляет из себя двухмерное поле, в котором отображаются ноты. Горизонтальная ось обозначает время от начала воспроизведения, вертикальная ось обозначает высоту ноты. Ноты, отмеченные разным цветом воспроизводятся разными виртуальными инструментами. Для удобства ввода и чтения нот в левой части добавлено изображение клавиатуры, в котором каждая клавиша находится напротив ряда нот соответствующей высоты. За отображение и обработку интерфейса экрана, содержащего PianoRollView отвечает класс EditorViewController, являющийся наследником класса UIViewController из фреймворка UIKit.

Было принято решение использовать SwiftUI для описания элемента PianoRollView, так как он позволяет описывать интерфейс декларативным методом, что лучше подходит для описания различных состояний редактора. Экран, содержащий PianoRollView написан при помощи UIKit, поскольку в данном случае императивное описание интерфейса позволяет более гибко настроить архитектуру интерфейса.

PianoRollView и EditorViewController общаются при помощи моделей PianoRollSettings, содержащий настройки редактора и PianoRollModel, содержащий описание звуков в проекте. При изменении звуков в PianoRollView в результете действий пользователя меняется модель PianoRollModel и EditorViewController получает информацию об изменениях.

PianoRollView может находиться в двух режимах – редактирование и выделение.

В режиме редактирования добавление нот, их перемещение и изменение длительности осуществляется нажатиями и жестами смахивания. Перемещение фокуса осуществляется с помощью жестов смахивания в области, свободной от нот.

В режиме выделения при перемещении пальца по экрану осуществляется выделение нот. Ноты, попавшие в выделенную подсвечиваются и записываются в модель selectedModel, которая в дальнейшем используется для воспроизведения или применении алгоритма. При перемещении пальца по экрану с выделенными нотами, поверх основного PianoRollView отрисовывается дополнительный прозрачный PianoRollView, содержащий только выделенные ноты для отображения перемещения выделенной области. При отпускании пальца дополнительный PianoRollView скрывается и выделенные ноты перемещаются в соответствии с новыми координатами.

За обработку, кодирование и воспроизведение полученных в редакторе нот отвечает класс EditorAudioInteractor. Для воспроизведения используется фреймворк AudioKit, который предоставляет движок AudioEngine, некоторые виртуальные инструменты, класс Sequensor, отвечающий за управление виртуальными инструментами. Для удобства работы вокруг виртуальных инструментов написаны классы-обёртки, позволяющие загружать в них ноты, полученные в редакторе, воспроизводить их и получать ноты, закодированные в формате MIDI для отправки на сервер или сериализации.

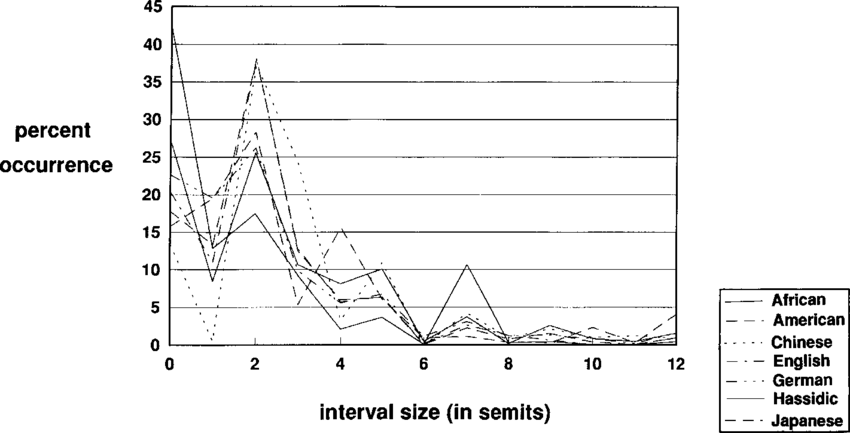
Для кодирования выделенной области, которая может содержать несколько инструментов, используется объект masterInstrument, который не воспроизводит звук при проигрывании, но содержит в себе все ноты проекта и возможность кодирования в MIDI.

* + 1. Алгоритм генерации вариации мелодии

Генерация мелодии осуществляется на серверной части приложения. Сервер получает мелодию в формате MIDI и преобразует в массив объектов MelodyNote, содержащих информацию о высоте, длительности ноты, позиции в треке и громкости. Также запрос может содержать лад, в котором будет происходить генерация. Если лад не был передан, то он определяется на основе полученных нот, используя метод derive фреймворка music21.

Генерация вариаций мелодии осуществляется при помощи генетического алгоритма. Мутация мелодии осуществляется при помощи перемещения случайных нот вниз и вверх по ладу или замены ноты на другую случайную ноту из начальной мелодии. Кроссинговер происходит при помощи соединении половин двух мелодий. Отбор и выбор лучшей мелодии осуществляется при помощи функции I + a \* D + b \* C + c \* R, где I – сумма обратной вероятности появления интервала в мелодии, D – сумма направлений движения мелодии с весами, R – диапазон между самой низкой и самой высокой ноты в мелодии, C – количество не консонантных интервалов среди нот, которые звучат одновременно. Параметр I позволяет оптимизировать шаги между соседними нотами, параметр D позволяет изменить среднюю длину восходящих и низходящих участков мелодии, C позволяет избежать диссонансов в одновременно играющих нотах R позволяет сохранить исходный диапазон.

Вероятности появления интервала в мелодии были получены из исследования [27].



(Рис. 2 – частота использования интервалов в музыке различных культур)

* + 1. Алгоритм генерации вариации ритма

Генерация ритма осуществляется на серверной части приложения. Сервер получает мелодию в формате MIDI и преобразует в массив объектов MelodyNote, содержащих информацию о высоте, длительности ноты, позиции в треке и громкости.

Генерация вариаций мелодии осуществляется при помощи генетического алгоритма. Мутация ритма осуществляется при помощи перемещения случайных нот вперёд и назад во времени на четверть или треть такта. Кроссинговер происходит при помощи соединении половин двух мелодий. Отбор и выбор лучшей мелодии осуществляется при помощи функции, подсчитывающей количество уникальных временных интервалов между соседними нотами. Такой подход позволяет получать более стабильный ритм.

* + 1. Алгоритм генерации вариации аккордов

Генерация аккордов осуществляется на серверной части приложения. Сервер получает мелодию в формате MIDI и преобразует в массив объектов MelodyNote, содержащих информацию о высоте, длительности ноты, позиции в треке и громкости. Также запрос может содержать лад, в котором будет происходить генерация. Если лад не был передан, то он определяется на основе полученных нот, используя метод derive фреймворка music21.

Генерация аккордов к мелодии осуществляется при помощи генетического алгоритма. Мутация последовательностей аккордов осуществляется при помощи замены случайных элементов на новые случайные аккорды из тональности, содержащие от 3 до 4 нот. Генерация случайных аккордов осуществляется при помощи выбора случайной ноты тональности и добавления третьих ступеней этой тональности, поскольку это самые распространённые аккорды, которые не лоамают гармонию. Кроссинговер в генетическом алгоритме осуществляется при помощи соединения половин последовательностей аккордов. Отбор и выбор лучшей последовательности происходит при помощи подсчёта диссонансов между нотами соседних аккордов, а также между аккордами и нотами начальной мелодии. Эти две величины складываются с определёнными весами. Такой подход позволяет создавать аккордовые последовательности, которые не диссонируют между собой и мелодией, а дополняют её.

* + 1. Алгоритм генерации вариации ритмического рисунка барабанов

Генерация ритмического рисунка барабанов осуществляется на серверной части приложения. Сервер получает мелодию в формате MIDI и преобразует в массив объектов MelodyNote, содержащих информацию о высоте, длительности ноты, позиции в треке и громкости.

Генерация происходит при помощи генетического алгоритма, по аналогии с генерацией ритма, но популяция ритмических рисунков инициализируется не из исходного ритма, а из популярных ритмических рисунков разных жанров. Также при отборе учитывается количество совпадений с ритмом исходной мелодии.

* + 1. Алгоритм транспонирования мелодии в другой лад

Транспонирование мелодии осуществляется на сервере.

Ноты полученной от клиента мелодии преобразуются в последовательность ступеней исходного лада, а затем в набор нот из целевого лада.

* + 1. Алгоритм сохранения проекта

Сохранение проекта происходит в мобильной части приложения.

Модель PianoRollModel преобразуется в объект json и сохраняется в файл на устройестве.

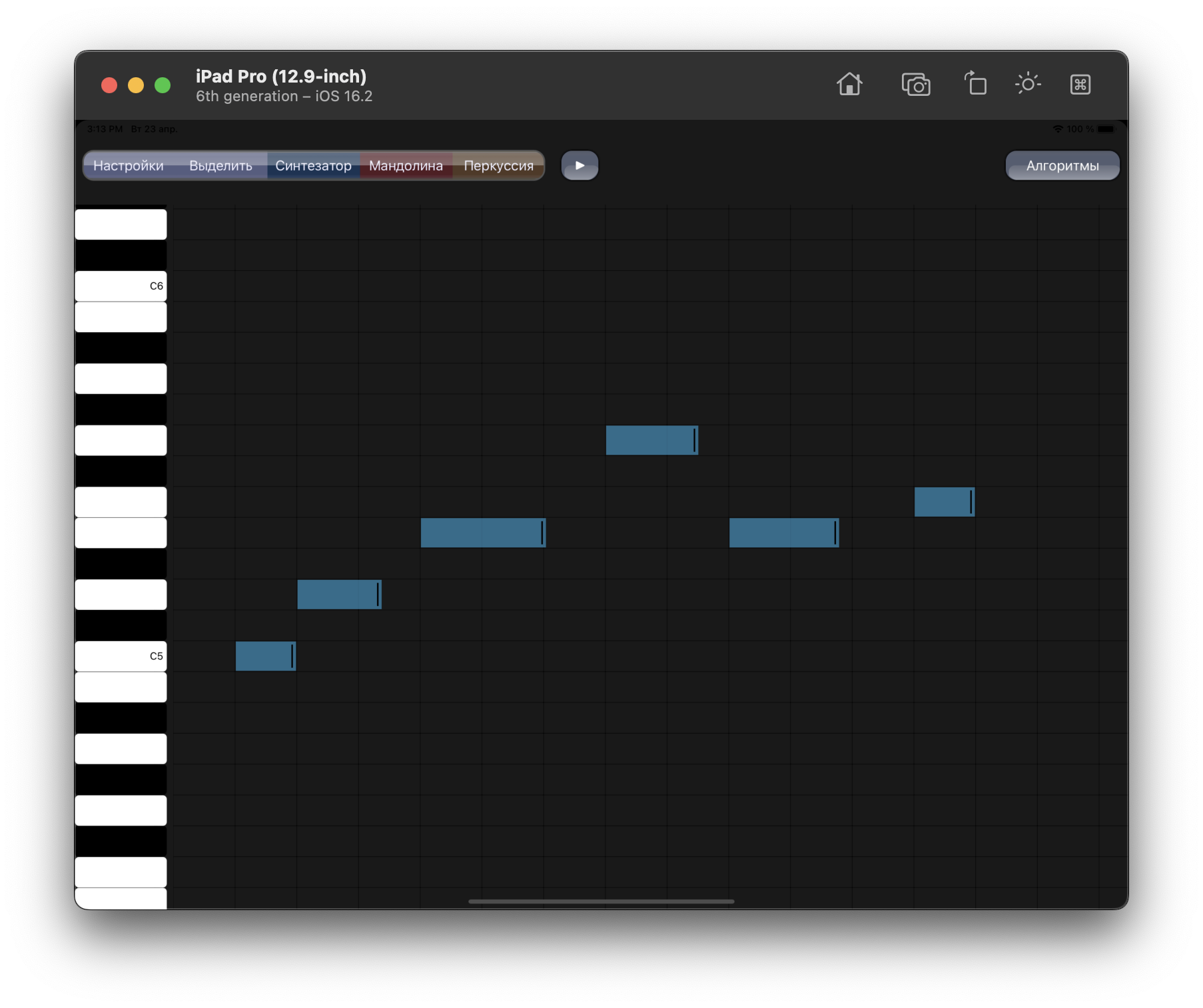
* + 1. Экспорт проекта

Экспорт проекта происходит в мобильной части приложения.

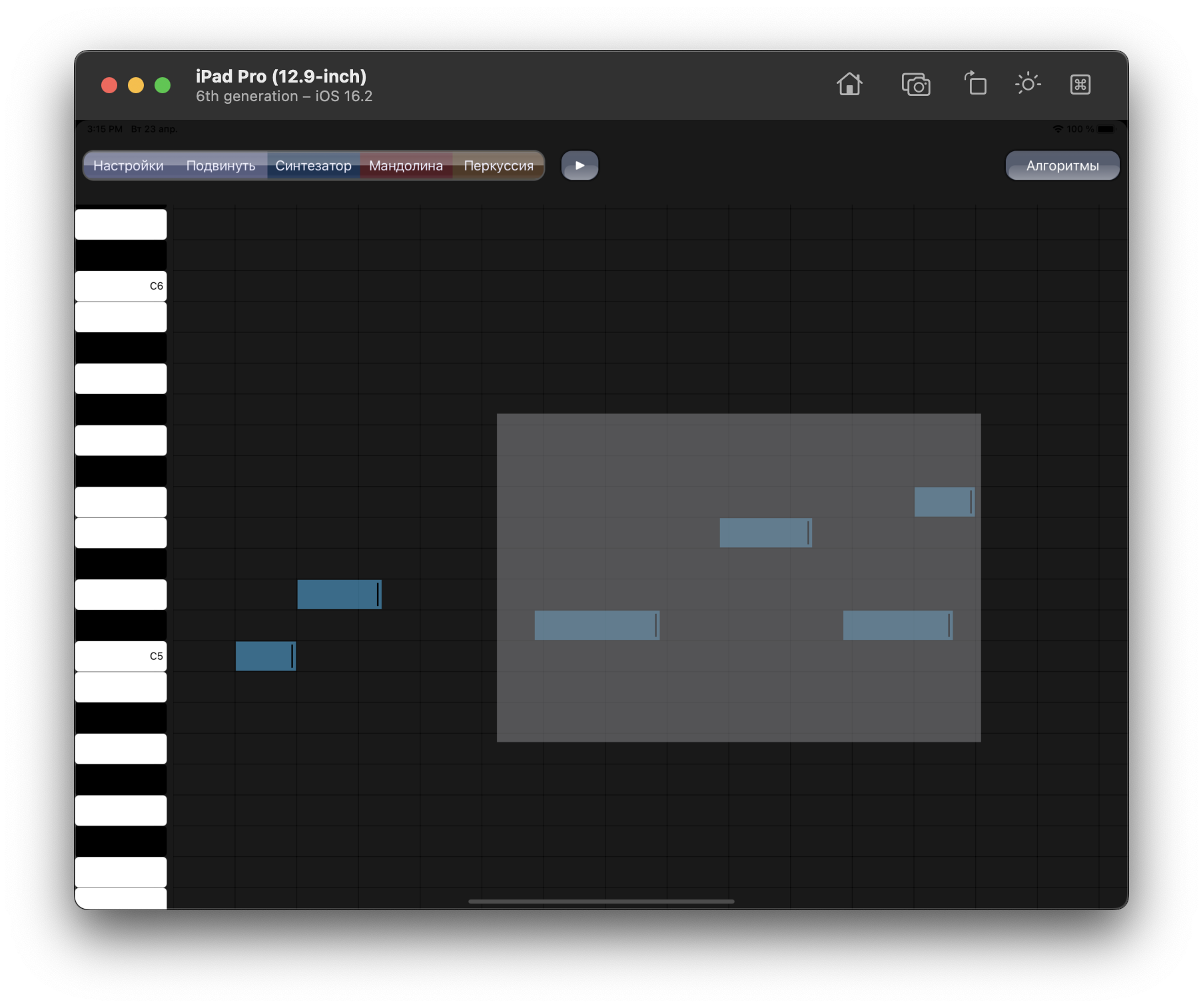
Ноты, предназначенные для воспроизведения различными инструментами загружаются в объект класса Sequencer в различные дорожки. После этого дорожки кодируются в файл в формате MIDI и файла сохраняется на устройстве.

# Требования к интерфейсу

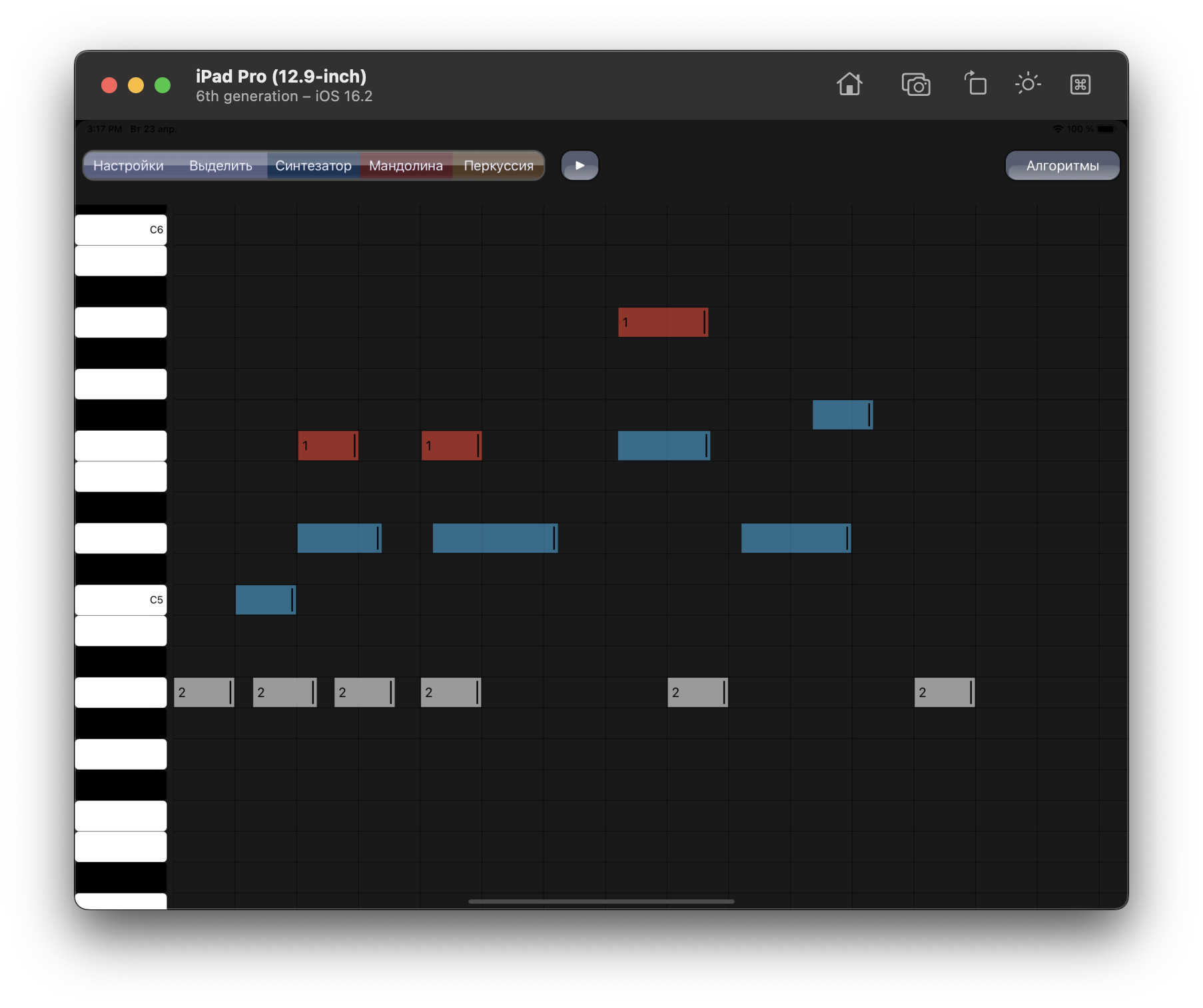
1. Регистрация и вход
   1. Поле для ввода электронной почты
   2. Поле для ввода пароля
   3. Кнопка войти/зарегистрироваться
2. Экран выбора проекта
   1. Прокручиваемый список проектов
      1. При нажатии на проект он открывается
      2. При долгом нажатии появляется возможность переименовать проект
   2. Кнопка создания проекта
3. Экран проекта
   1. Прокручиваемый экран с нотами (рис. 3)
   2. Кнопки изменения инструмента, который проигрывает выделенную область (рис. 5)
   3. Кнопка начать/остановить запись
   4. Возможность выделить фрагмент записи (рис. 4)
   5. Возможность применить к фрагменту один из алгоритмов синтеза аранжировки
      1. Список алгоритмов
      2. Поля ввода параметров алгоритмов (Рис. 6)
   6. Возможность удалить ноту
   7. Возможность добавить ноту
   8. Кнопка экспорта в MIDI



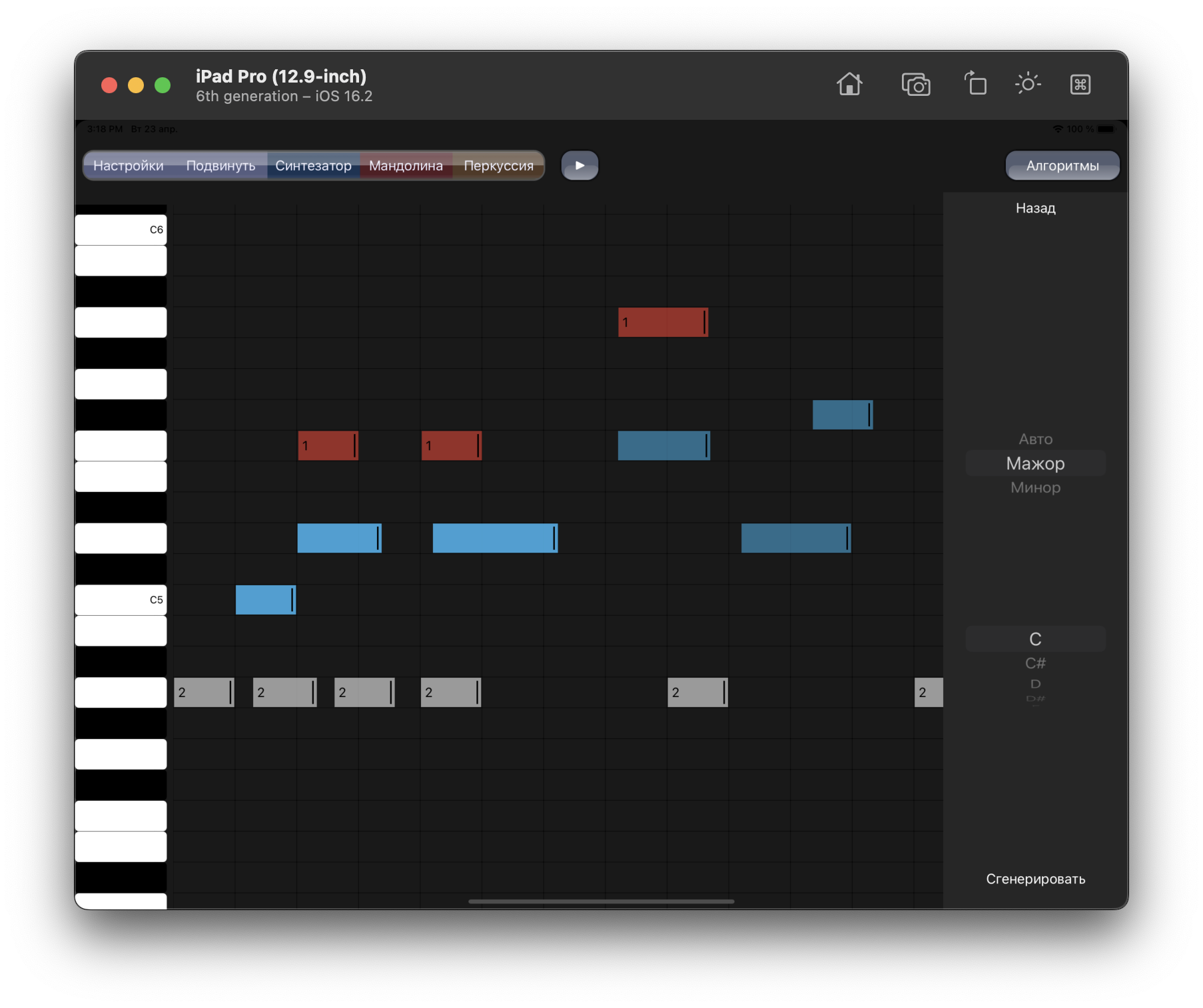
(Рис. 3 – экран редактирования проекта)



(Рис. 4 – выделение области)



(Рис. 5 – редактирование инструментов)



(Рис. 6 – генерация мелодии)

# Требования к надежности

Программа должна обеспечивать проверку корректности входных данных.

# Требования к обеспечению надёжного (устойчивого) функционирования программы

Надежное функционирование программы должно быть обеспечено состоянием сервера, на котором будет развернуто приложение:

1. Организацией питания технических средств;
2. Отсутствием на техническом устройстве различных видов вирусных программ;
3. Наличие постоянного доступа к сети интернет;
4. Регулярной проверкой оборудования и программного обеспечения на наличие сбоев и неполадок.

# Время восстановления после отказа

Если отказ был вызван какими-либо внешними факторами, например, сбоем электропитания, и при этом не произошел непоправимый̆ сбой операционной системы, то время восстановления не должно превышать времени, требующегося на перезагрузку операционной системы и запуск программы.

Если отказ был вызван неисправностью технических средств или непоправимым сбоем операционной системы, то время восстановления не должно превышать времени, необходимого для устранения неисправностей технических и программных средств.

# Отказы из-за некорректных действий оператора

Устройство API должно исключать некорректные действия оператора: все опасные ситуации должны быть обработаны и протестированы.

# Условия эксплуатации

# Климатические условия эксплуатации

Требования к климатическим условиям эксплуатации не предъявляются.

# Требования к видам обслуживания

Обслуживание не требуется.

# Требования к численности и квалификации персонала

1. Минимальное количество персонала, требуемого для работы программы – 1 человек: пользователь приложения;
2. Пользователь программы должен быть ознакомлен с тематикой программного приложения;
3. Пользователь должен знать основными музыкальные термины и обладать базовыми знаниями чтения нот с листа;

# Требования к составу и параметрам технических средств

Сервер:

1. Наличие оборудования для подключения к сети Internet
2. Сервер должен иметь доступ в сеть интернет с внешним ip адресом;
3. Минимальная тактовая частота процессора — 2 ГГц;
4. Рекомендуется 8 Гб ОЗУ или более;
5. Сервер должен обрабатывать 1000 запросов в секунду;
6. Минимальное дисковое пространство 20 Гб.

Клиент:

1. Наличие оборудования для подключения к сети Internet
2. Устройство iPhone или iPad с операционной системой IOS 16 и выше.
3. Минимальное дисковое пространство 20 Гб.

# Требования к информационной и программной совместимости

# Требования к информационным структурам и методам решения.

Не предъявляются.

# Требования к исходным кодам и языкам программирования

Исходный код серверной части должен быть написан на языке программирования Python с использованием библиотек Flask, music21.

Исходный код клиентской части должен быть написан на языке программирования Swift с использованием библиотеки UIKit и SwiftUI для создания интерфейса.

# Требования к программным средствам, используемым программой

Для работы программы необходим следующий состав программных средств: Docker.

# Требования к маркировке и упаковке

Программа поставляется в виде ссылки на репозиторий на Github, в котором должны содержаться исходный код программы и программная документация.

Программное изделие должно иметь маркировку с обозначением наименования изделия, темы разработки, фамилии, имени и отчества исполнителя и руководителя разработки, учебной группы и года выпуска изделия.

# Требования к транспортировке и хранению

Программа хранится в репозитории на Github по ссылке.

Специальные требования к транспортировке не предъявляются.

# ТРЕБОВНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

# Состав программной документации

* «Программа для синтеза аранжировок музыкальных композиций». Техническое задание (ГОСТ 19.201-78 [7]);
* «Программа для синтеза аранжировок музыкальных композиций». Программа и методика испытаний (ГОСТ 19.301-79 [8]);
* «Программа для синтеза аранжировок музыкальных композиций». Текст программы (ГОСТ 19.401-78 [9]);
* «Программа для синтеза аранжировок музыкальных композиций». Пояснительная записка (ГОСТ 19.404-79 [10]);
* «Программа для синтеза аранжировок музыкальных композиций». Руководство оператора (ГОСТ 19.505-79 [11]);

# Специальные требования к программной документации

* Документы к программе должны быть выполнены в соответствии с ГОСТ 19.106-78 [6] и ГОСТами к каждому виду документа (см. п. 5.1.);
* Пояснительная записка должна быть загружена в систему Антиплагиат через LMS «НИУ ВШЭ». Лист, подтверждающий загрузку пояснительной записки, сдается в учебный офис вместе со всеми материалами не позже, чем за день до защиты ВКР;
* Документация и программа также сдается в электронном виде в формате .pdf или .docx. в архиве формата .zip или .rar;
* Все документы перед защитой курсовой работы должны быть загружены в информационно-образовательную среду НИУ ВШЭ LMS (Learning management system) в личном кабинете, дисциплина - «Подготовка к защите ВКР», одним архивом.

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

# Ориентировочная экономическая эффективность

В рамках темы выпускной квалификационной работы в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия» Национального исследовательского университета «Высшая школа экономики», факультет компьютерных наук, департамент программной инженерии экономическая эффективность не предусмотрена.

Несмотря на это, существует возможность реализовать данный проект на коммерческой основе. Например, при помощи публикации приложения на платформе “App Store”.

# Предполагаемая потребность

Данная программа будет интересна людям, которые пишут собственные композиции как в качестве хобби, так и в рамках профессиональной деятельности.

# Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

На момент начала разработки аналогами программы являются:

1. Autochords [16]
2. Chord Suggester [17]
3. Cthulhu [18]
4. AIVA [19]
5. Amadeus Code [20]

Сравнительная таблица аналогов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Autochords | Chord Suggester | Cthulhu | AIVA | Amadeus Code |  |
| Генерация аккордов | + | + | + | + | + | + |
| Генерация ритмических рисунков | - | - | +- | + | + | + |
| Генерация аранжировки | - | - | - | + | - | + |
| Результат формулизовав в виде, понятном человеку | + | + | + | - | + | + |
| Распознание мелодии с помощью микрофона | - | - | - | - | - | + |
| Экспорт в MIDI | - | + | + | + | + | + |
| Генерация на основе данной мелодии | - | - | + | - | - | + |
| Воспроизведение результата | + | + | + | + | + | + |

# ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

# Виды испытаний

Проверка программного продукта, в том числе и на соответствие техническому заданию, осуществляется согласно «Программе и методике испытаний», а также пункту 5.2. Программа и методика испытаний (ГОСТ 19.301-79 [8]).

Защита выполненного проекта осуществляется комиссии, состоящей из преподавателей департамента программной инженерии, в утверждённые приказом декана ФКН сроки.

# Общие требования к приемке работ

Прием программы будет утвержден при корректной работе программы в соответствии с пунктом 4.1.1 при различных входных данных, соответствующих условиям в пункте 4.1.2 данного документа и при предоставлении полной документации к продукту, указанной в пункте 5.1, выполненной в соответствии с требованиями, указанными в пункте 5.2 данного технического задания.

# ПРИЛОЖЕНИЕ 1

**ТЕРМИНОЛОГИЯ**

Таблица 1 – Терминология

|  |  |
| --- | --- |
| Аккорд | одновременное сочетание трёх и более музыкальных звуков разной высоты (точнее, разных высотных классов), воспринимаемое слухом как целостный элемент звуковысотной вертикали. |
| Арпеджио | способ исполнения аккордов, при котором звуки аккорда берутся последовательно один за другим |
| Гармония | в данном документе - звуковысотная организация музыки — как многоголосной, так и одноголосной. |
| Лад | последовательность звуков, расположенных по высоте в восходящем или нисходящем порядке. Отдельные звуки в таком ряду в теории музыки именуются ступенями. |
| IOS | мобильная операционная система для смартфонов, электронных планшетов, носимых проигрывателей, разрабатываемая и выпускаемая американской компанией Apple. |
| MIDI | стандарт цифровой звукозаписи на формат обмена данными (интерфейс) между электронными музыкальными инструментами. |
| Мелодия | один голос музыкальной фактуры, который трактуется в теории музыки и непосредственно воспринимается слухом как композиционно-техническое целое. |
| Музыкальная композиция (далее - композиция) | категория музыковедения и музыкальной эстетики, характеризующая предметное воплощение музыки в виде выработанного и завершённого в себе музыкального произведения. |
| Аранжировка | модификация, адаптация, изменение нотного текста музыкального произведения для исполнения его иным, чем в оригинале, составом инструментов |
| Нота | графическое обозначение музыкальных звуков. Также используется как синоним к музыкальным звукам. |
| Интервал | соотношение двух музыкальных звуков по их высоте. |

**ПРИЛОЖЕНИЕ 2**

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. ГОСТ 19.603-78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
13. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
14. RESTful API / [Электронный ресурс] // URL: https://aws.amazon.com/ru/what-is/restful-api/ (дата обращения - 17 ноября 2023)
15. Swagger UI / [Электронный ресурс] // URL: https://swagger.io/tools/swagger-ui/ (дата обращения - 17 ноября 2023)
16. AutoChords / [Электронный ресурс] // URL: https://autochords.com/ (дата обращения - 17 ноября 2023)
17. Suggester / [Электронный ресурс] // URL: <https://apps.apple.com/us/app/suggester-chords-and-scales/id504740787> (дата обращения - 17 ноября 2023)
18. Cthulhu / [Электронный ресурс] // URL: https://xferrecords.com/products/cthulhu (дата обращения - 17 ноября 2023)
19. AIVA / [Электронный ресурс] // URL: https://www.aiva.ai/ (дата обращения - 17 ноября 2023)
20. Amadeus Code / [Электронный ресурс] // URL: https://amadeuscode.com/en/ (дата обращения - 17 ноября 2023)
21. Apple Developer Documentation / [Электронный ресурс] // URL: https://developer.apple.com/documentation/ (дата обращения - 20 ноября 2023)
22. Music Notation / [Электронный ресурс] // URL: https://github.com/music-notation-swift/music-notation (дата обращения - 17 ноября 2023)
23. Beethoven / [Электронный ресурс] // URL: https://github.com/vadymmarkov/Beethoven/tree/master (дата обращения - 17 ноября 2023)
24. Python / [Электронный ресурс] // URL: https://docs.python.org/3/ (дата обращения - 17 ноября 2023)
25. Flask / [Электронный ресурс] // URL: https://flask.palletsprojects.com/en/3.0.x/ (дата обращения - 17 ноября 2023)
26. MIDI / [Электронный ресурс] // URL: <https://ru.wikipedia.org/wiki/MIDI> (дата обращения - 17 ноября 2023)
27. David Huron , “Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles”, 2001 / [Электронный ресурс] // URL: <https://www.researchgate.net/figure/Frequency-of-occurrence-of-melodic-intervals-in-notated-sources-for-folk-and-popular_fig6_215646542> (дата обращения - 20 апреля 2024)
28. Dorien Herremans, Ching-Hua Chuan, Elaine Chew, “Functional Taxonomy of Music Generation Systems” 2017. [Электронный ресурс]. // URL: <https://dl.acm.org/doi/abs/10.1145/3108242>
29. S. A. Hedges, "Dice music in the eighteenth century", Music Lett., vol. 59, pp. 180-187, 1978.
30. F. P. Brooks, A. L. Hopkins, P. G. Neumann and W. V. Wright, "An experiment in musical composition", *IRE Trans. Electron. Comput.*, vol. EC-5, pp. 175-182, Sep. 1956.
31. Генетический алгоритм [Электронный ресурс] // URL: <https://ru.wikipedia.org/wiki/Генетический_алгоритм> (дата обращения - 23 апреля 2024)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |