

Федеральное государственное автономное  
образовательное учреждение высшего образования

Университет ИТМО

Дисциплина: Экономика программной инженерии

## **Лабораторная работа 1**

Вариант <https://www.muztorg.ru/>

**Выполнили:**

Ляо Ихун

Васильков Александр Сергеевич

**Группа:** Р34131

**Преподаватель:**

Машина Екатерина Алексеевна

2023 г.

Санкт-Петербург

## **Наивный метод**

Шаг №	Функционал	Оценка мин./ чел.час	Оценка сред./ чел.час	Оценка макс./ чел.час
<b>Анализ</b>				
1.1	Анализ предоставляемого функционала (расставление приоритетов)	6	8	12
1.2	Анализ сайтов-конкурентов	5	10	24
1.3	Оценка (себе) стоимости/затрат	3	4	5
1.4	Согласование требований и сроков реализации	24	36	48
	Итого	38	58	89
<b>Проектирование</b>				
2.1	Выбор технологий frontend(Подбор нужного фреймворка, настройка)	8	12	16
2.2	Выбор технологий backend(Продумывание архитектуры и взаимодействия необходимых компонентов)	12	18	24
2.3	Отрисовка макета	24	48	56
	Итого	44	78	96
<b>Разработка</b>				

Frontend (Создан на React)				
3.1	Создание таблицы для отрисовки товаров	3	5	9
3.2	Навигация через каталог и его отрисовка	12	36	48
3.3	Верхняя “панель” (Услуги,помощь...)	1	2	3
3.4	Адаптивная верстка, плавные анимации, удобное отображение таблиц	16	24	36
3.5	Фильтр для таблиц + интеграция с каталогом	15	24	30
3.6	Отрисовка других элементов на главной странице сайта	6	12	18
3.7	Вход и регистрация	10	15	20
	Итого	63	118	164
Backend				
4.1	Создание БД кластера, настройка, создание таблиц индексов	50	65	80
4.2	Создание логики авторизации	30	45	60
4.3	Создание логики фильтрации и сортировки товаров	60	70	85
4.4	Создание логики заказа товара	30	45	60

4.5	Создание логики корзины	20	30	50
		190	225	335
Тестирование				
5.1	Модульное тестирование	48	72	96
5.2	Интеграционное тестирование	12	20	24
5.3	Функциональное тестирование	48	72	96
5.4	Системное тестирование	30	45	60
	Итого	138	209	276
Конфигурирование				
6.1	Аренда хостинга	8	24	40
6.2	SSL сертификат	4	8	9
6.3	выбор окружения	12	18	24
6.4	проверка настройка параметров окружения	5	10	15
	Итого	29	60	88
Итого:		502	778	1048

## ***PERT метод***

Шар№	Функционал	Оценка мин./ чел.час	Оценка сред./ чел.час	Оценка макс./ чел.час	$E_i = \frac{(P_i + O_i + 4M_i)}{6}$	$CKO_i = \frac{(P_i - O_i)}{6}$
Анализ						
1.1	Анализ предоставляемого функционала (расставление приоритетов)	6	8	12	8,33	1,0
1.2	Анализ сайтов-конкурентов	5	10	24	11,5	3,17
1.3	Оценка (себе) стоимости/затрат	3	4	5	4,0	0,33
1.4	Согласование требований и сроков реализации	24	36	48	36,0	4,0
Проектирование						
2.1	Выбор технологий frontend(Подбор нужного фреймворка, настройка)	8	12	16	12,0	1,33

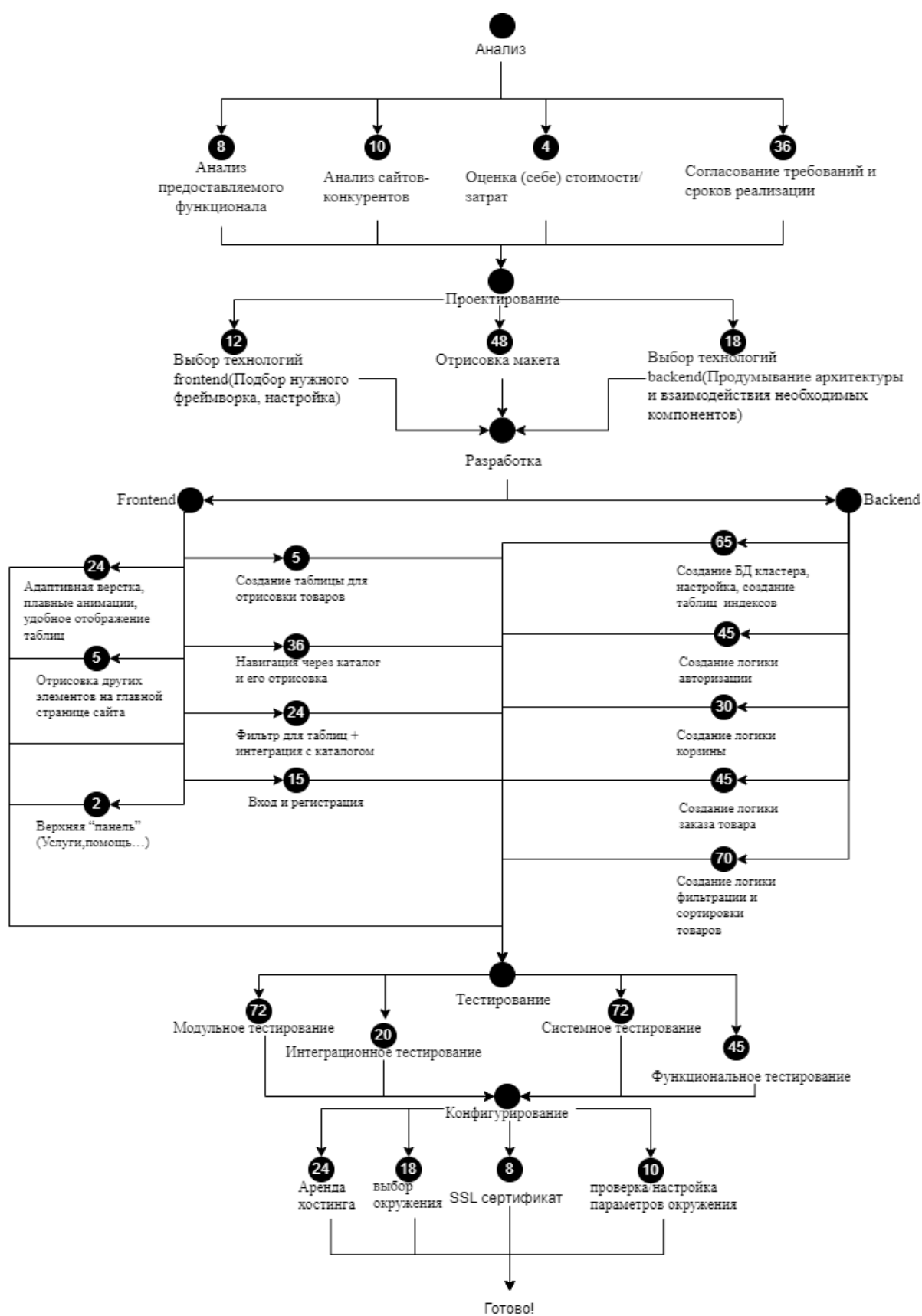
2.2	Выбор технологий backend(Продумывание архитектуры и взаимодействия необходимых компонентов)	12	18	24	18,0	2,0
2.3	Отрисовка макета	24	48	56	45,33	5,33
Разработка						
Frontend (Создан на React)						
3.1	Создание таблицы для отрисовки товаров	3	5	9	5,33	1,0
3.2	Навигация через каталог и его отрисовка	12	36	48	34,0	6,0
3.3	Верхняя "панель" (Услуги,помощь ...)	1	2	3	2,0	0,33
3.4	Адаптивная верстка, плавные анимации, удобное отображение таблиц	16	24	36	24,67	3,33
3.5	Фильтр для таблиц +	15	24	30	23,5	2,5

	интеграция с каталогом					
3.6	Отрисовка других элементов на главной странице сайта	6	12	18	12,0	2,0
3.7	Вход и регистрация	10	15	20	15,0	1,67
Backend						
4.1	Создание БД кластера, настройка, создание таблиц индексов	50	65	80	65,0	5,0
4.2	Создание логики авторизации	30	45	60	45,0	5,0
4.3	Создание логики фильтрации и сортировки товаров	60	70	85	70,83	4,17
4.4	Создание логики заказа товара	30	45	60	45,0	5,0
4.5	Создание логики корзины	20	30	50	31,67	5,0
Тестирование						
5.1	Модульное тестирование	48	72	96	72,0	8,0
5.2	Интеграционное тестирование	12	20	24	19,33	2,0

5.3	Функциональное тестирование	48	72	96	72,0	8,0
5.4	Системное тестирование	30	45	60	45,0	5,0
Конфигурирование						
6.1	Аренда хостинга	8	24	40	24,0	5,33
6.2	SSL сертификат	4	8	9	7,5	0,83
6.3	выбор окружения	12	18	24	18,0	2,0
6.4	проверка настройка параметров окружения	5	10	15	10,0	1,67
$E = \sum E_i = 777$ $CKO = \sqrt{\sum CKO_i^2} = 20.8$ $E_{95\%} = E + 2 * CKO = 798.8$						



## Метод критического пути



**Критический путь:**

$8+4+36+12+48+18+5+36+24+15+5+24+65+45+30+45+70+72+20+24+18+8+10=642$

**Длинный путь: 784**

**Выполнение проекта:** при ориентире на минимальное время разработки получаем, что для выполнения нам необходимо

**Команда:**

- 1x Аналитик
- 2x Frontend-разработчик
- 2x Backend-разработчик
- 1x DevOps

**Рабочий день считаем:** 6 часов + 1ч обед + 1ч тех. перерыв

**Аналитику потребуется:** 41ч работы или 7 рабочих дней

**Команде backend-разработчиков потребуется:** 255ч или 42 рабочих дня

**Команде frontend-разработчиков потребуется:** 117ч или 20 рабочих дня

**Двум командам для тестирования потребуется:** 92ч или 8 рабочих дней

**DevOps для конфигурации и настройки итогового проекта потребуется:** 60ч или 10 рабочих дней

**Рассчитаем время разработки и общее время для завершения проекта:**

Frontend и Backend можно делать параллельно, после чего уже делать общее интеграционное тестирование.

В конце после всех настроек получаем:  $41+255+117+92+60=565$  часов или **63 рабочих дня.**

## Метод функциональных точек

При анализе методом функциональных точек надо выполнить следующую последовательность шагов:

1. Определение типа оценки
2. Определение области оценки и границ продукта
3. Подсчет функциональных точек, связанных с данными
4. Подсчет функциональных точек, связанных с транзакциями
5. Определение суммарного количества не выровненных функциональных точек (UFP)
6. Определение значения фактора выравнивания (FAV)
7. Расчет количества выровненных функциональных точек (AFP)

## Определение типа оценки

Продукт. Оценивается объем уже существующего и установленного продукта.

## Определение области оценки и границ продукта

Все функции. Рассчитываем все необходимые (реально используемые), а не дополнительные или только основные функции. Границы системы определены на UseCase диаграмме.

№	Название	RET	DET	Сложность	UFP
1	Личный кабинет	Личные данные	Фамилия, имя, никнейм, email, телефон, день рождения, специализация, город(7)	Low	7
2	Форма регистрации	данные для регистрации	Имя, Фамилия, Телефон, E-mail, Пароль, Дата рождения, Регион(7)	Low	7
3	Заказ обратного звонка	контактные данные, комментарий, время для звонка	имя, телефон, дата и время звонка, комментарий(4)	Low	7
4	оформление заказа	контактные данные, получения заказа, способ оплаты	Телефон, имя, фамилия, email, магазин, способ оплаты(6)	Low	7

## Подсчет функциональных точек, связанных с транзакциями

Транзакция - это элементарный неделимый замкнутый процесс, представляющий значение для пользователя и переводящий продукт из одного консистентного состояния в другое.

- **EI** (external inputs) — внешние входные транзакции, элементарная операция по обработке данных или управляющей информации, поступающих в систему из вне.
- **EO** (external outputs) — внешние выходные транзакции, элементарная операция по генерации данных или управляющей информации, которые выходят за пределы системы. Предполагает определенную логику обработки или вычислений информации.
- **EQ** (external inquiries) — внешние запросы, элементарная операция, которая в ответ на внешний запрос извлекает данные или управляющую информацию.
- **FTR** (file type referenced) — позволяет подсчитать количество различных файлов (информационных объектов) модифицируемых, или считываемых в транзакции.
- **DET** (data element type) — неповторяемое уникальное поле данных. Примеры. EI: поле ввода, кнопка. EO: поле данных отчета, сообщение об ошибке. EQ: поле ввода для поиска, поле вывода результата поиска.

Чем отличаются транзакции:

Функция	Тип транзакции		
	EI	EO	EQ
Изменяет поведение системы	Основная	Дополнительная	-
Поддержка одного или более внутренних логических файлов	Основная	Дополнительная	-
Представление информации пользователю	Дополнительная	Основная	Основная

№	Название	Тип	FTR	DET	Сложность	UFP
1	Форма обратной связи	EI	1	4	Low	3
2	Форма регистрации	EI	1	7	Low	3
3	Просмотр профиля	EQ	1	7	Low	3
4	Поиск	EQ	2	1	Low	3
5	Просмотр подборок	EO	1	1	Low	3

6	Просмотр статических страниц	EI	1	1	Low	3
7	Просмотр информации о товаре	EO	3	1	Low	4

### Определение суммарного количества не выровненных функциональных точек (UFP)

**UFP** =22+32=54

№	Параметр	Вес (DI)
1	Обмен данными	2
2	Распределенная обработка данных	0
3	Производительность	0
4	Ограничения по аппаратным ресурсам	0
5	Транзакционная нагрузка	0
6	Интенсивность взаимодействия с пользователем	2
7	Эргономика	2
8	Интенсивность изменения данных	1
9	Сложность обработки	0
10	Повторное использование	2
11	Удобство инсталляции	0
12	Удобство администрирования	2
13	Портируемость	1
14	Гибкость	0
$TDI = \sum DI = 12$ $VAF = (TDI * 0.01) + 0.65 = 0.77$		

## Расчет количества выровненных функциональных точек (AFP)

$$AFP = UPF \times VAF = 54 * 0.77 = 41.58$$

Метод анализа функциональных точек ничего не говорит о трудоемкости разработки оцененного продукта. Вопрос решается просто, если компания разработчик имеет собственную статистику трудозатрат на реализацию функциональных точек. Если такой статистики нет, то для оценки трудоемкости и сроков проекта можно использовать метод COCOMO II.

## COCOMO II

### Оценка размера программного продукта в KSLOC

#### Стек технологий:

- React
- Backend

Разделим функциональность между слоями:  $\frac{3}{4}$  - frontend и  $\frac{1}{4}$  - backend. Подсчитаем размер по KSLOC:

[Таблица коэффициентов](#)

$$KSLOC = UFP * SIZE = 54 * 0.75 * 0.047 + 54 * 0.25 * 0.053 = 2.619$$

### Оценка уровней факторов масштаба

- PREC - прецедентность, наличие опыта аналогичных разработок
- FLEX - гибкость процесса разработки
- RESL - архитектура и разрешение рисков
- TEAM - слаботанность команды
- PMAT - зрелость процессов

Название фактора	Уровень фактора	Значение уровня
PREC	High	2.48
FLEX	High	2.03
RESL	Low	5.65
TEAM	High	2.19
PMAT	Very Low	7.80

### Оценка уровней множителей трудоемкости

Для предварительной оценки проекта необходимо оценить уровень семи множителей трудоемкости M:

- PERS - квалификация персонала
- RCPX - сложность и надежность продукта
- RUSE - разработка для повторного использования
- PDIF - сложность платформы разработки
- PREX - опыт персонала
- FCIL - оборудование
- CSER - требуемое выполнение графика р

Название	Уровень	Значение
PERS	Nominal	1.00
RCPX	Very Low	0.60
RUSE	Low	0.95
PDIF	Low	0.87
PREX	High	0.87
FCIL	Nominal	1.00
CSER	Nominal	1.00

### Оценка трудоемкости проекта

$$PM = A \times SIZE^E \times \prod_{i=1}^n EM_i \quad A = 2.94$$

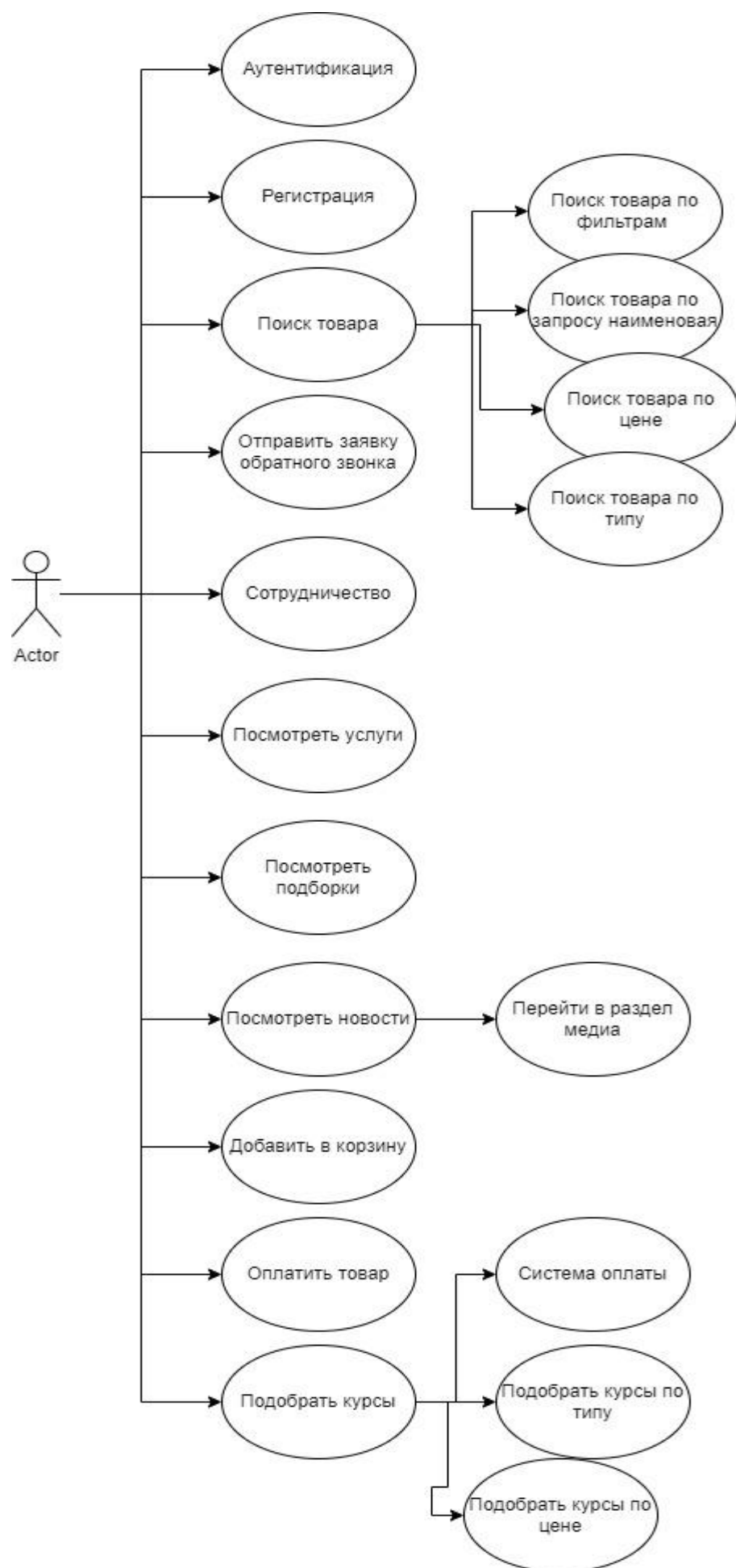
$$E = B + 0.01 \times \sum_{j=1}^5 SF_j \quad B = 0.91$$

- SIZE — размер продукта в KSLOC
- $EM_i$  — множители трудоемкости
- $SF_j$  — факторы масштаба
- $n=7$  — для предварительной оценки
- $n=17$  — для детальной оценки

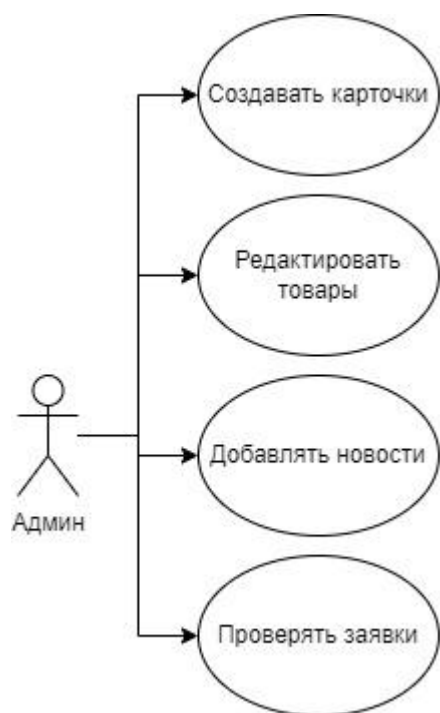
$$E = 0.91 + 0.01 \times (2.48 + 2.03 + 5.65 + 2.19 + 7.80) = 1.1115$$

$$PM = 2.94 \times 2.619^{1.1115} \times (1.00 \times 0.60 \times 0.95 \times 0.87 \times 0.87 \times 1.00 \times 1.00) = 3.71 \text{ ч./мес} \\ = 594 \text{ ч./ч.}$$

Use Case диаграмма для пользователя:







### Оценка веса прецедентов

Сложность	Вес(UUCW)	Количество	Затраты
Low	5	20	100
Medium	10	5	50
High	15	6	90
<b>Нескорректированный вес варианта использования (UUCW)</b>			240

### Оценка веса акторов

Сложность	Вес(UAW)	Количество	Затраты
Low	1	2	2
Medium	2	0	0

High	3	2	6
<b>Масса актера без корректировки (UAW)</b>			8

### Оценка веса технических факторов

<b>Фактор</b>	<b>Вес (W)</b>	<b>Номинальная стоимость(F)</b>	<b>Затраты</b>
Распределённость	1	0	0
Производительность	2	3	6
Эффективность для пользователя	3	4	12
Сложная внутренняя обработка	1	0	0
Повторное использование кода	2	1	2
Простота установки	1	1	1
Простота использования	3	3	9
Переносимость	1	0	0
Простота изменений	3	5	15
Многопоточность	1	0	0
Дополнительные возможности безопасности	1	1	1
Доступ к другим системам	1	2	2
Необходимы тренажеры для пользователей	1	0	0
<b>Общий технический фактор (TFactor)</b>			48
<b><math>TCF = 0.6 + (TF/100)</math></b>			1.08

## Оценка веса факторов окружения

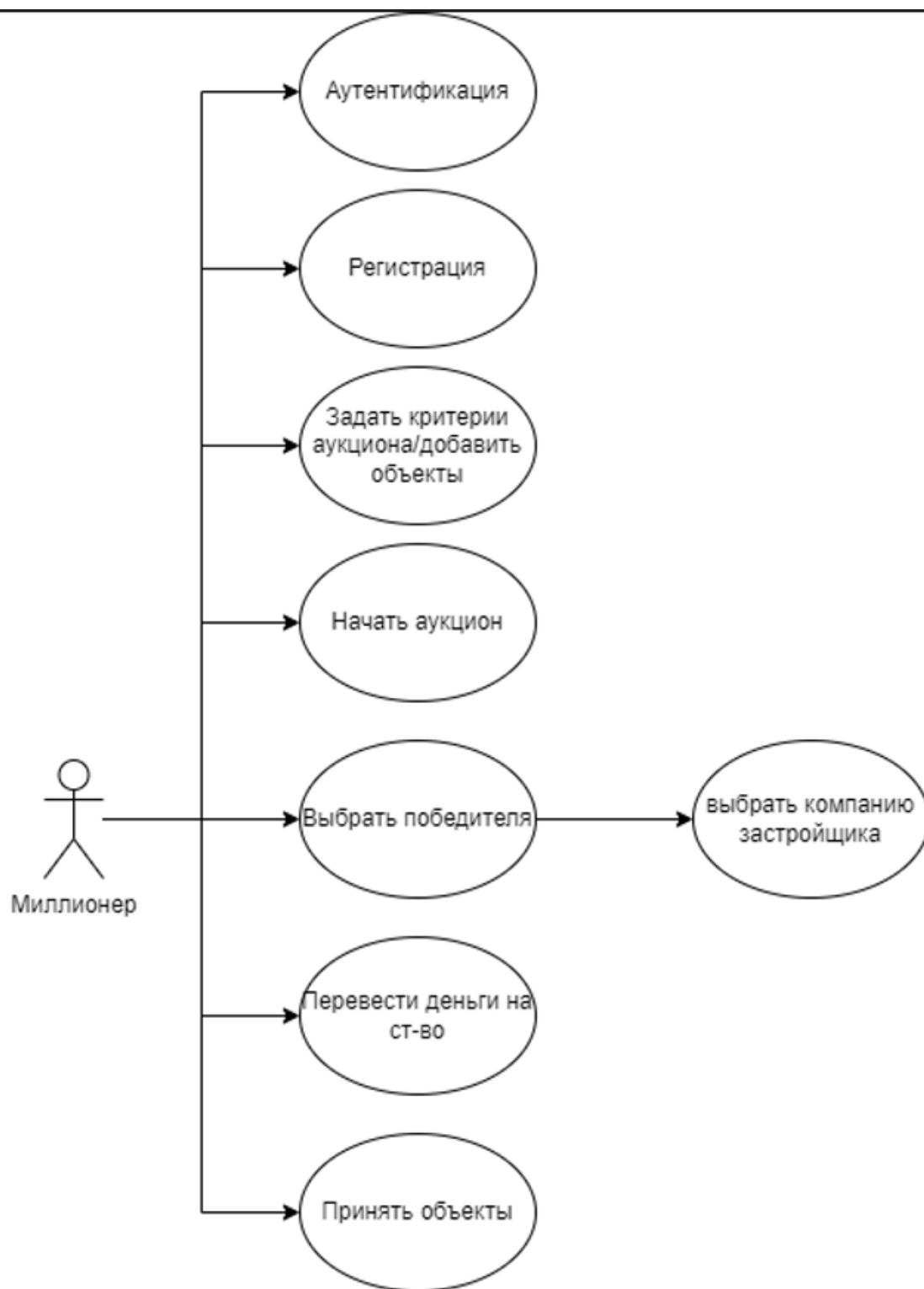
Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Знаком с моделью проекта, которая используется	1.5	4	6.0
Опыт применения	0.5	3	1.5
Опыт в веб разработке	1.0	4	4.0
Возможность ведущего аналитика	0.5	2	1.0
Мотивация	1.0	2	2.0
Стабильные требования	1.5	2	1.5
Частичная занятость	-1.0	3	-3.0
Сложность языка программирования	-1.0	4	-4.0
<b>Общий фактор окружающей среды (EFactor)</b>			9.0
<b>ECF = 1.4 + (-0.03 * EF)</b>			1.13

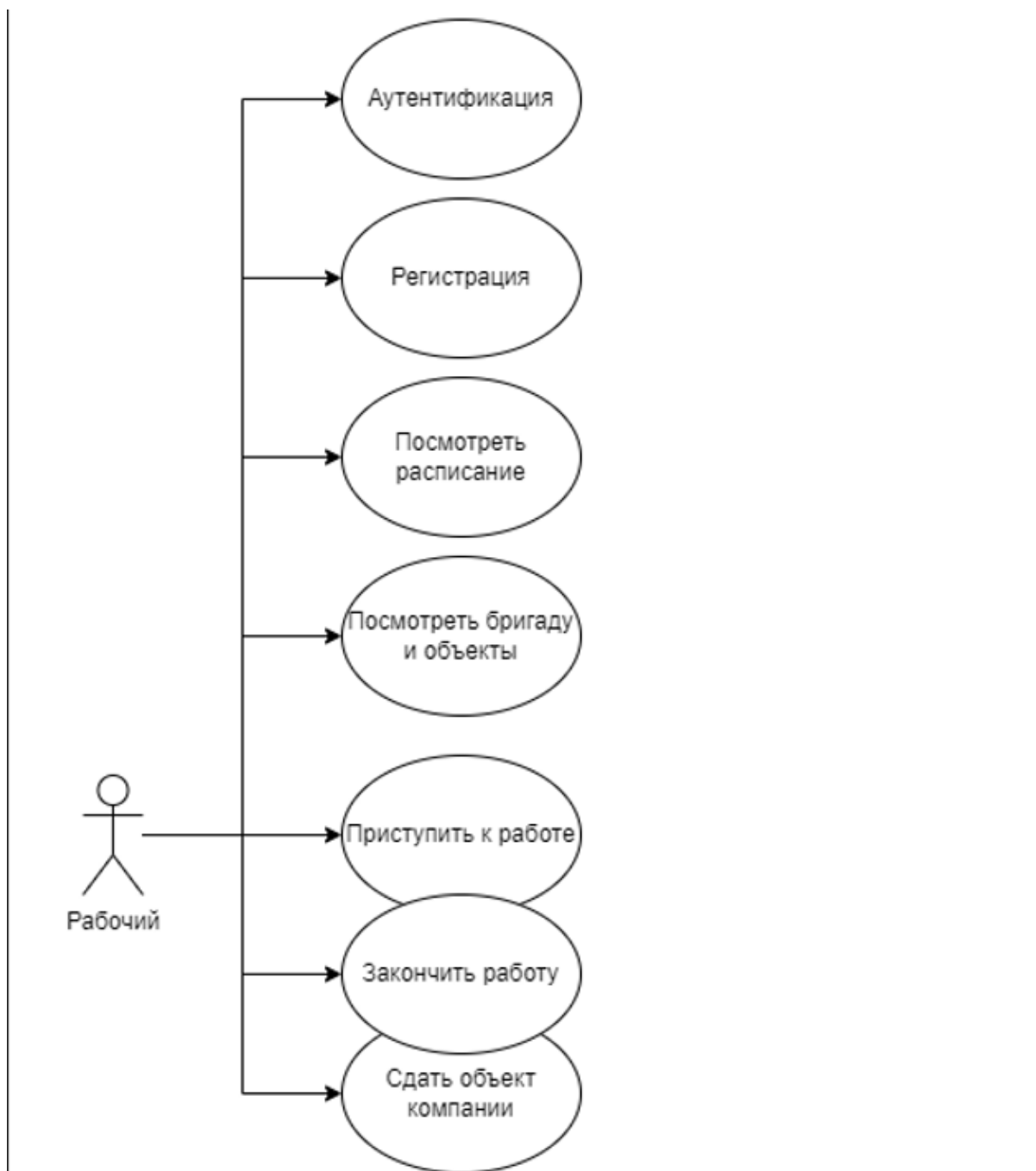
Подсчет UCP

$$UCP' = (UCW + UAW) * TCF * ECF = 303$$

Подсчёт фактора продуктивности (PF) на основе прошлого проекта

В качестве примера мы выбрали проект по **ИСБД(курсовая работа)**, в которую входит выполнение всех лабораторных работ для двух человек.





#### Оценка веса прецедентов

Сложность	Вес (UUCW)	Количество	Затраты
Low	5	7	35
Medium	10	8	80
High	15	0	0

<b>Нескорректированный вес варианта использования (UUCW)</b>	115
--	-----

### Оценка веса акторов

<b>Сложность</b>	<b>Вес (AUW)</b>	<b>Количество</b>	<b>Затраты</b>
Low	1	1	1
Medium	2	0	0
High	3	1	3
<b>Масса актера без корректировки (UAW)</b>			4

### Оценка веса технических факторов

<b>Фактор</b>	<b>Вес (W)</b>	<b>Номинальная стоимость(F)</b>	<b>Затраты</b>
Распределённость	2.0	0	0.0
Производительность	1.0	1	1.0
Эффективность для пользователя	1.0	2	2.0
Сложная внутренняя обработка	1.0	1	1.0
Повторное использование кода	2.0	1	2.0
Простота установки	0.5	1	0.5
Простота использования	0.5	1	0.5
Переносимость	2.0	1	2.0
Простота изменений	2.0	3	6.0
Многопоточность	1.0	1	1.0
Дополнительные возможности безопасности	1.0	1	1.0
Доступ к другим системам	1.0	1	1.0

Необходимы тренажеры для пользователей	1.0	1	1.0
<b>Общий технический фактор (TFactor)</b>			19
<b>TCF = 0.6 + (TF/100)</b>			0.79

### Оценка веса факторов окружения

Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Знаком с моделью проекта, которая используется	1.5	2	3
Опыт применения	0.5	2	1
Опыт в веб разработке	1.0	5	5
Возможность ведущего аналитика	0.5	0	0
Мотивация	1.0	1	1
Стабильные требования	1.5	2	3
Частичная занятость	-1.0	3	-3
Сложность языка программирования	-1.0	4	-4
<b>Общий фактор окружающей среды (EFactor)</b>			15
<b>ECF = 1.4 + (-0.03 * EF)</b>			0.95

### Подсчет UCP

$$UCP' = (UCW + UAW) * TCF * ECF = 75$$

### Подсчет трудоемкости проекта:

Предыдущая работа была выполнена за 35 часов на 2-ух человек

$$PF = E/UCP = 0.94$$

UCP = 303 - для нашего сайта

$$E = PF * UCP = 285 \text{ ч/ч} + \text{юр работа} + \text{работа с партнерами} + \text{заполнение всех данных} = 285 + 250 + 160 + 40 = 735 \text{ ч/ч}$$

## Анализ результатов

Метод	Затрат
Наивный	778
PERT	798
Критического Пути	565
COCOMO II	594
UCP	735

В результате мы видим, что USP, наивный и PERT показывают практически идентичные результаты, что с одной стороны может показывать, ошибку в оценке сложности проекта, который был выдан на лабораторную работу, так и переоценка сложности курсовой работы. На наш взгляд COCOMO II и Критического Пути имеют почти одинаковое значение из-за внесения малого кол-во транзакций в первом методе, что также отдалило ожидаемый нами результат от наиболее вероятного.

## Вывод

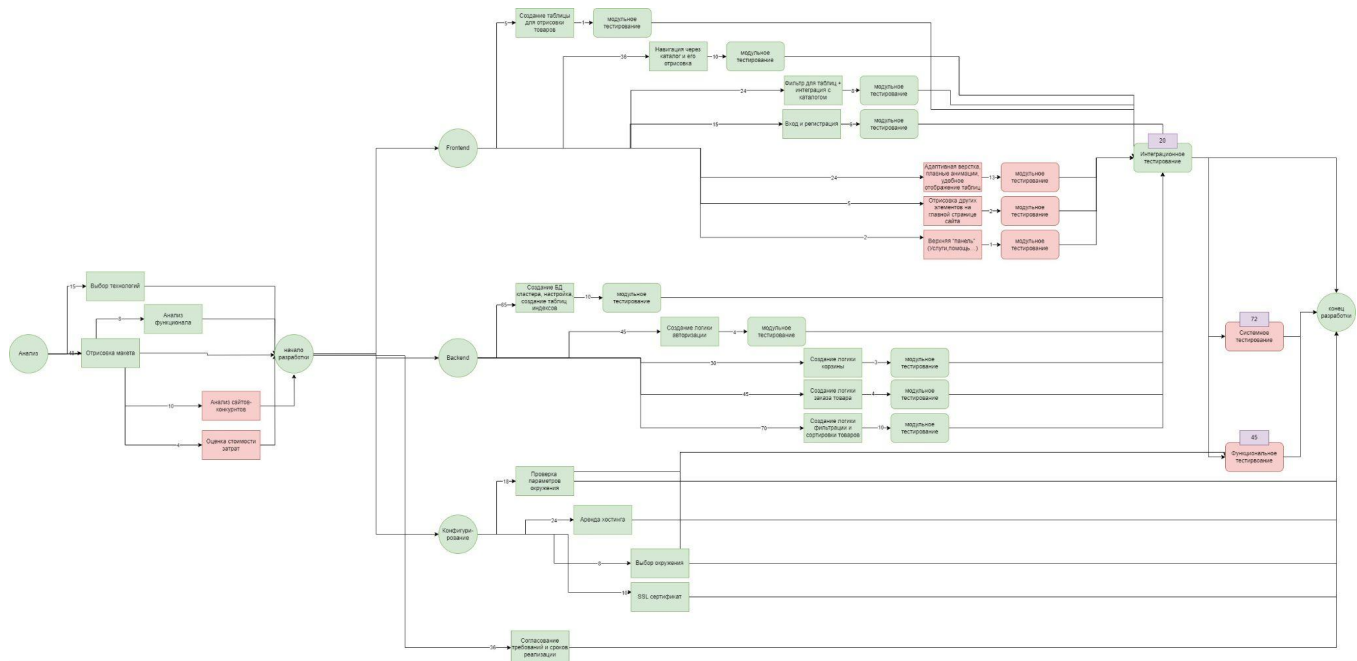
В процессе выполнения лабораторной работы мы примерили на себе роль менеджера, пытающегося адекватно выделить функции требуемого проекта и оценить время-затратность его выполнения. Ощутили, что оценивание одна из непростых задач, с которыми приходится сталкиваться в сфере программирования, как и реалистичные сроки выполнения.

---

### Сетевая диаграмма new!

---





Было предложено заменить предложенную нами команду, на команду fullstack-разработчиков, также совместить их работу с действиями аналитика и devops`а, для ускорения проектирования, аналитики и тестирования. Также был пересмотрен процесс разработки в котором многие процессы могут идти параллельно.

## Предположим, что у нас есть

### Команда:

- 4x Fullstack-разработчика

**Рабочий день считаем:** 6 часов + 2ч (в среднем) не функциональных

**Команде потребуется:**

- Этап 1 - Аналитика
  - Самое трудное это отрисовка макета, занимающая 48 ч, если выделить на нее 3 человека и параллельно 1 на две другие задачи, то потребуется 19ч работы или 4 рабочих дня.
- Этап 2 - Разработка
 

Для начала поставим везде по 1му разработчику, чтобы все задачи начали выполняться одновременно.

  - **Согласование требований.** Тогда через 6 дней один из них освободится, и присоединится к конфигурационному-треку.
  - **Конфигурация** Через 9 дней конфигурация будет настроена и к frontend-треку присоединится еще два разработчика.
  - **Frontend** Через 12 дней фронтенд будет полностью готов и протестирован, к backend-треку присоединится еще 3 разработчика.
  - **Backend** Тогда все 4 разработчика завершат backend через 21 день.
  - Еще день им понадобится на интеграционное тестирование.

**Рассчитаем время разработки и общее время для завершения проекта:**  
В конце после всех настроек получаем: **156 рабочих часа** или **26 рабочих дня**.

Федеральное государственное автономное  
образовательное учреждение высшего образования

Университет ИТМО

Дисциплина: Экономика программной инженерии

## **Лабораторная работа 2**

Вариант <https://www.muztorg.ru/>, kanban

**Выполнили:**

Ляо Ихун

Васильков Александр Сергеевич

**Группа:** Р34131

**Преподаватель:**

Машина Екатерина Алексеевна

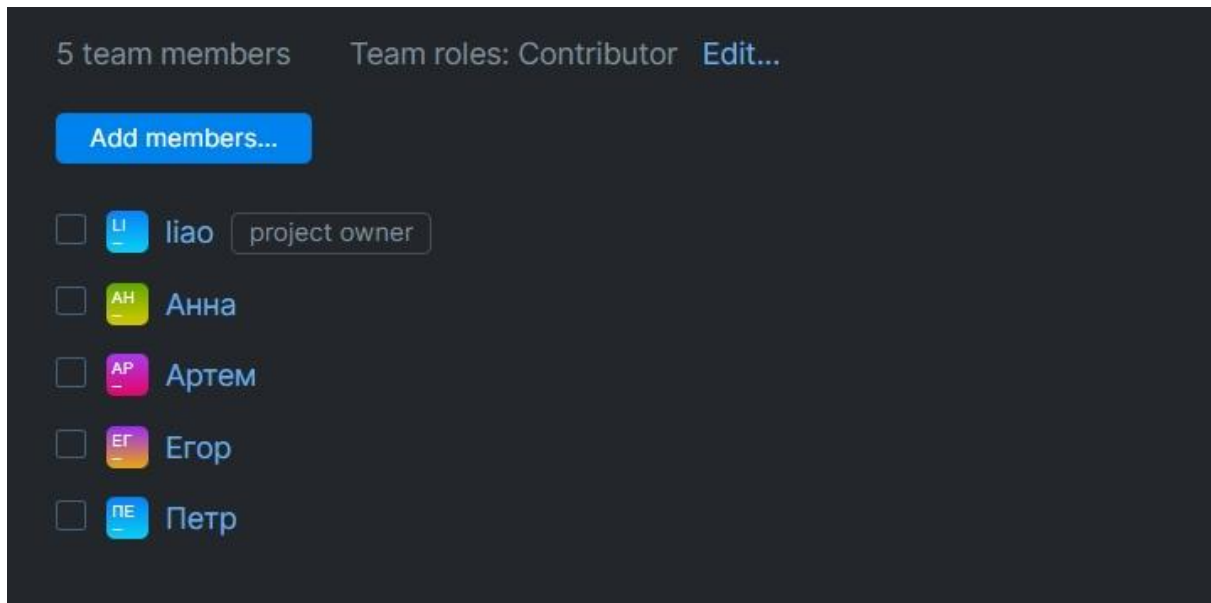
## Задание

Зарегистрироваться для использования бесплатной облачной версии ПО [YouTrack](#) для управления своим программным проектом:

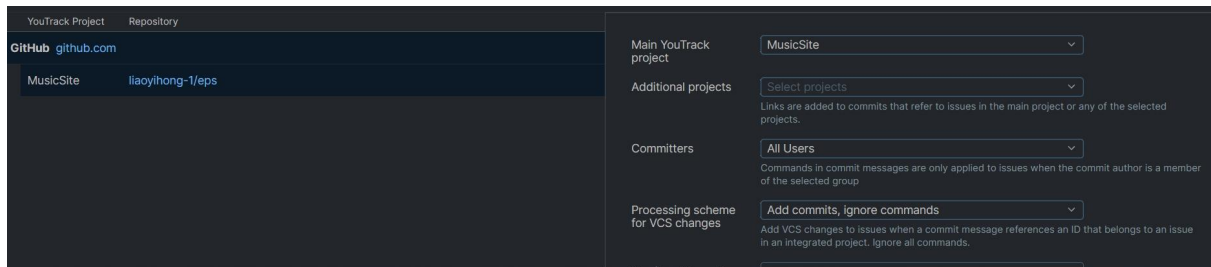
1. Создать учётные записи для всей своей проектной команды.
2. Интегрировать YouTrack с git репозиторием.
3. Настроить интеграцию с электронной почтой.
4. Создать проект с заданным в варианте профилем kanban.
5. Настроить столбцы доски для своего проекта.
6. Создать план работ над проектом и зафиксировать его в YouTrack (создать спринты, релизы и задачи, а также необходимые для работы ветви в репозитории).
7. Симулировать процесс разработки проекта, постепенно закрывая "выполненные" задачи и открывая новые.
8. После завершения снять метрики проекта и предоставить отчет, содержащий описание процесса конфигурации и настройки, описание выбранного workflow, и сформированные с помощью YouTrack отчеты, отражающие статистику работы над проектом. Обязательно должны быть приведены: отчет по исполнителям, burndown-диаграмма, отчет по времени, диаграмма Ганта.

## Выполнение

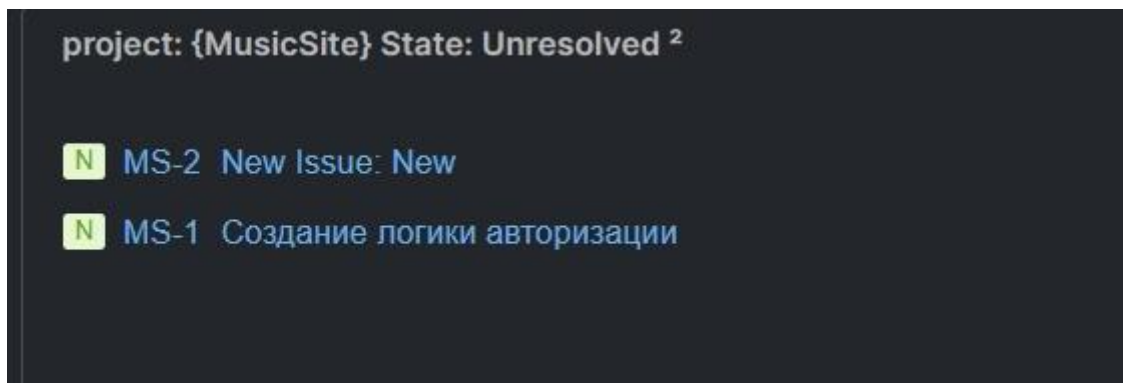
### Этап 1. Создать учётные записи для всей своей проектной команды



### Этап 2. Интегрировать YouTrack с git репозиторием



Используем issue с id MS-2 для тестирования интеграции:



## Commit changes



### Commit message

#MS-2 test github integration

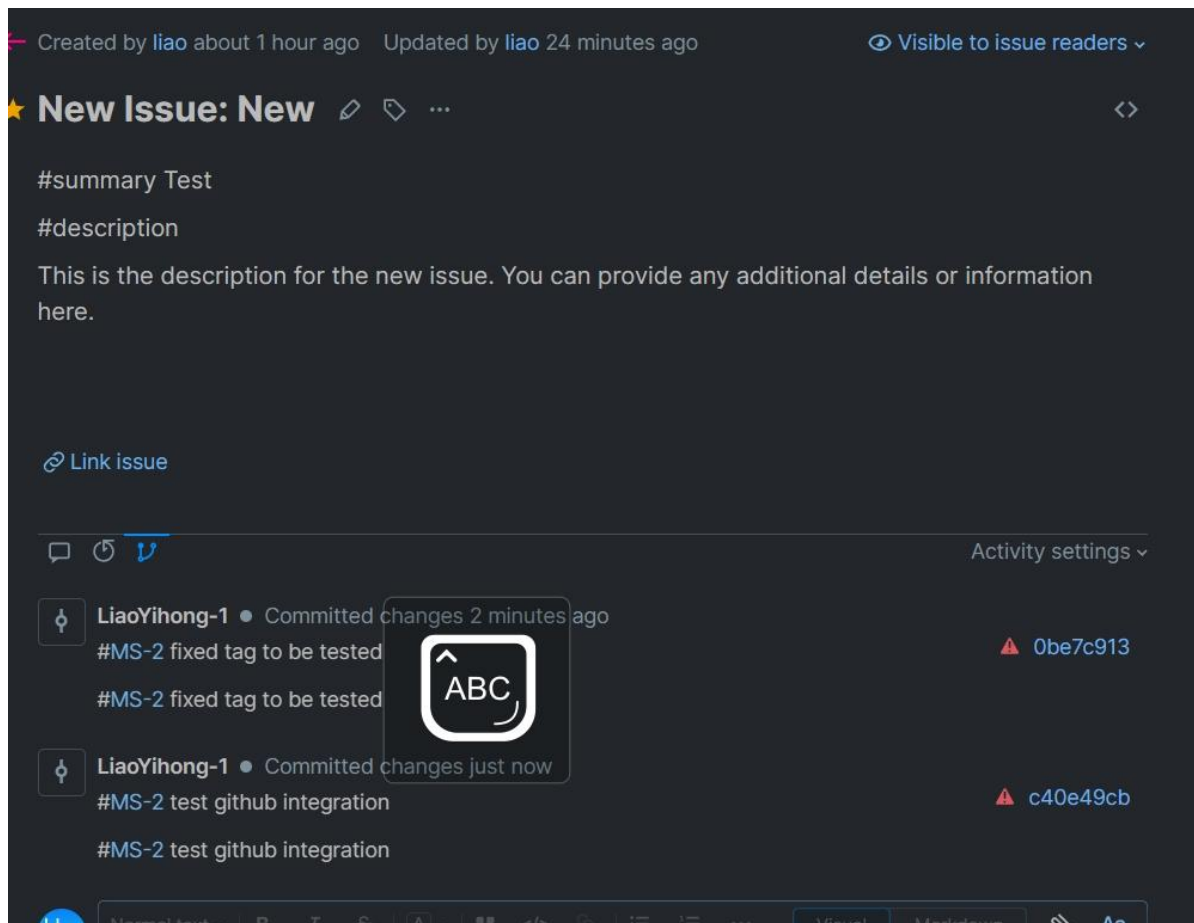
### Extended description

#MS-2 test github integration

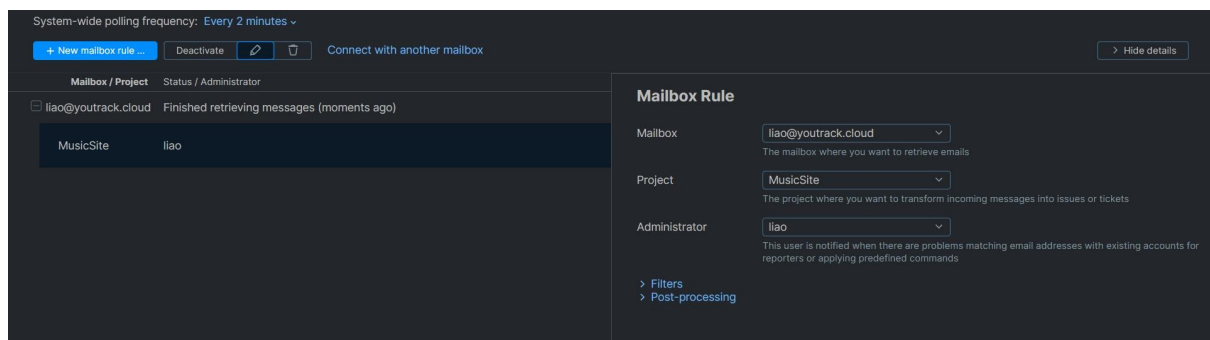
- ☒ Commit directly to the `main` branch
- ☐ Create a **new branch** for this commit and start a pull request  
[Learn more about pull requests](#)

Cancel

Commit changes



### Этап 3. Настроить интеграцию с электронной почтой



Отправим почту для создания issue:

## [Music] New Issue: New



yihong liao <x1761270349@gmail.com>

18:56

收件人: liao@youtrack.cloud

#summary Test

#description

This is the description for the new issue. You can provide any additional details or information here.

Запрос автоматически разработан:

## MS-2 New Issue: New



Music site <service@youtrack.cloud>

18:58



收件人: liao

##- Please enter your reply above this line -##

Dear liao,

From your message we automatically created the issue [MS-2](#) in our issue-tracking system

To edit or comment on the issue or watch its progress, you may need to [log into the tracker](#)

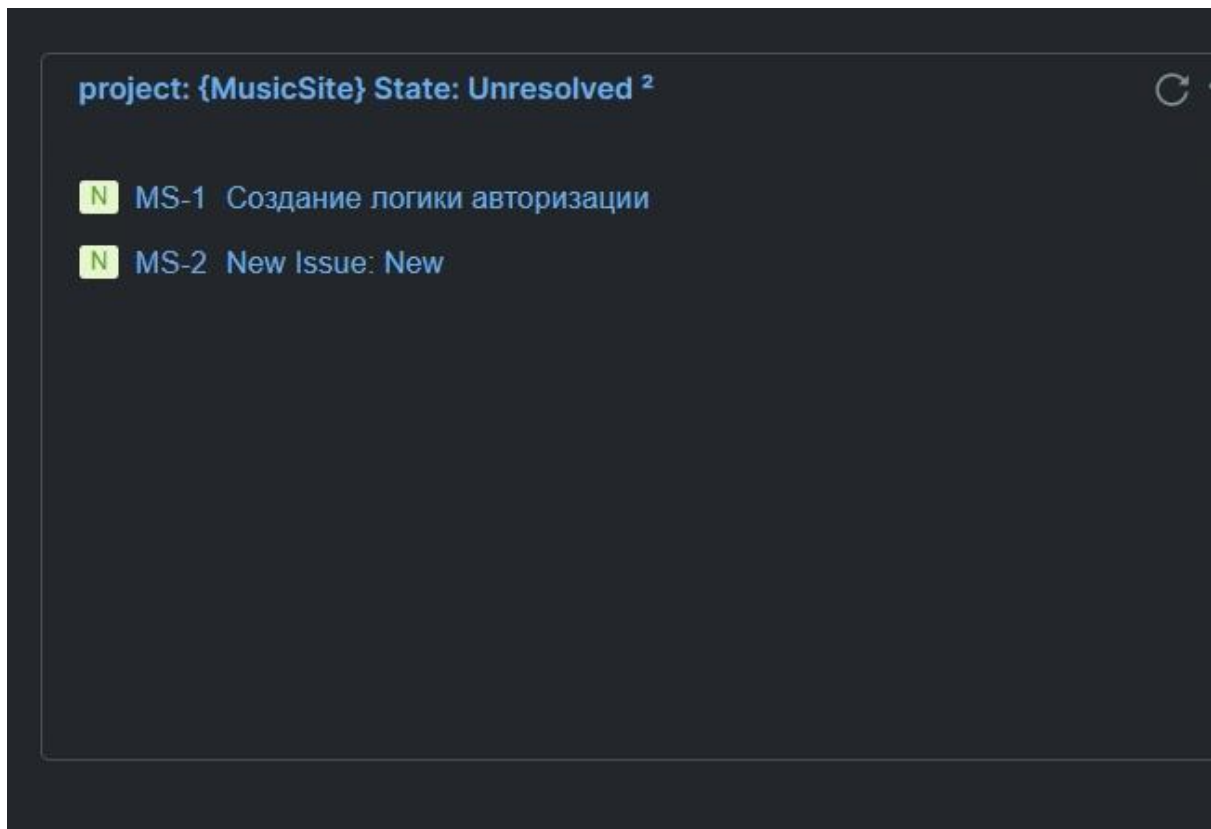
As a **reporter** of the issue, you will receive further notifications about any updates for your issue. All notifications will be sent to your e-mail address registered for the account. If you do not want to receive notifications, you can disable them in your user [profile](#).

From : [x1761270349@gmail.com](mailto:x1761270349@gmail.com)

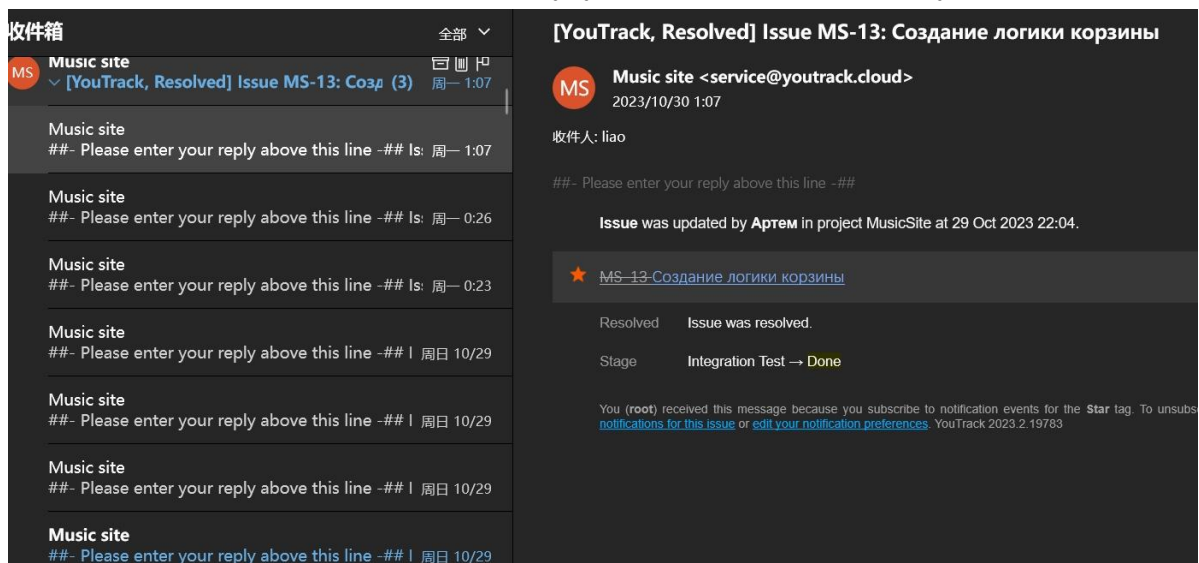
Sent : Fri Oct 27 2023 15:56:26 UTC

Новый issue создан:

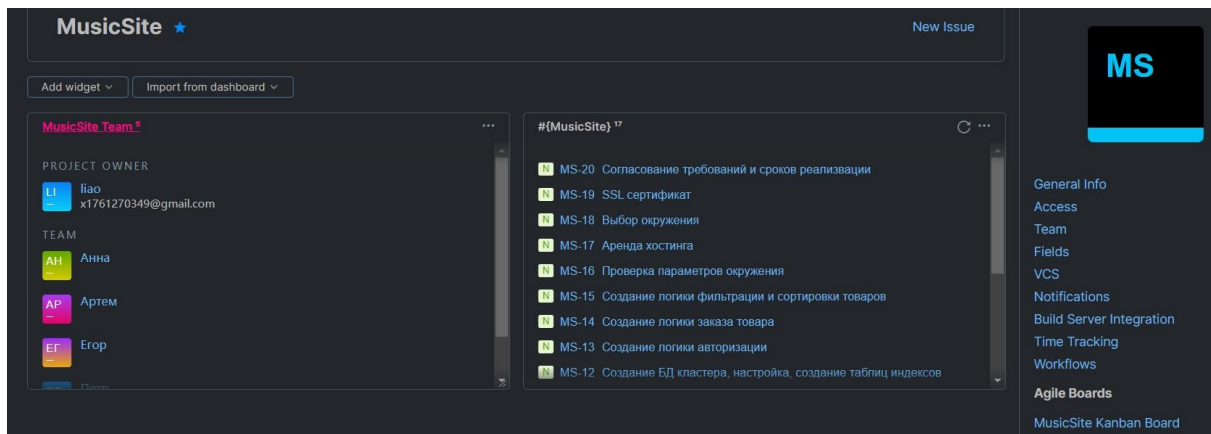




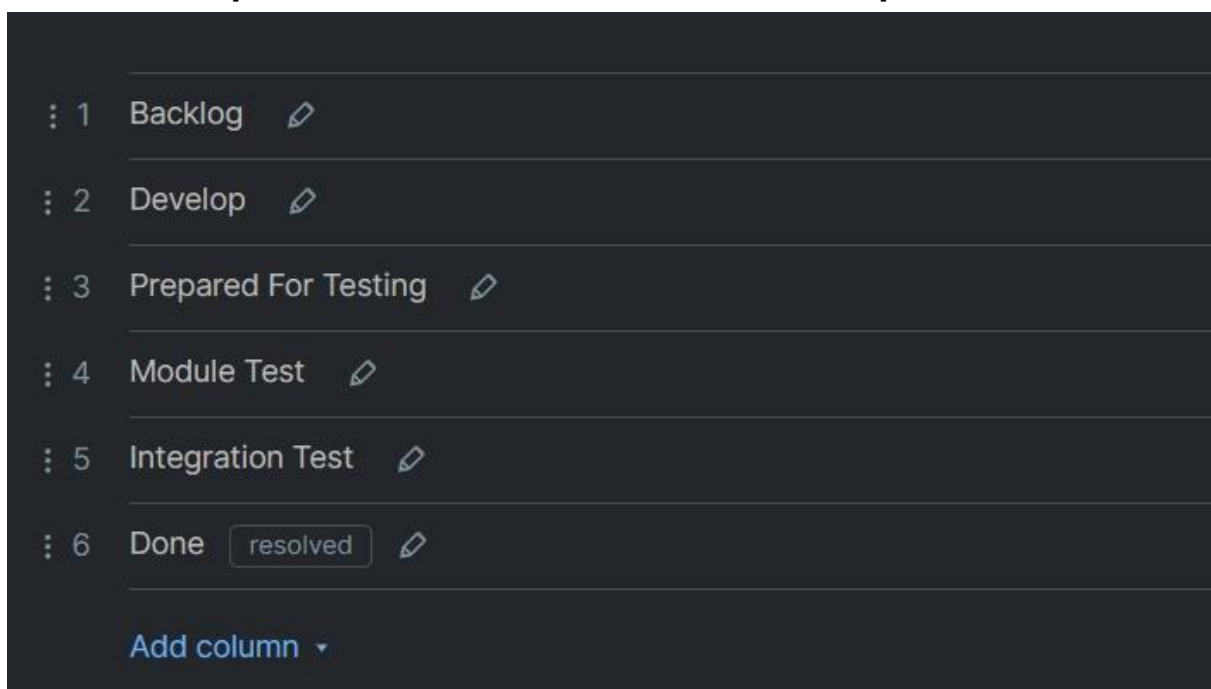
Потом изменения состояния issue тоже будут отправлены через почту.



**Этап 4. Создать проект с заданным в варианте профилем Kanban.**

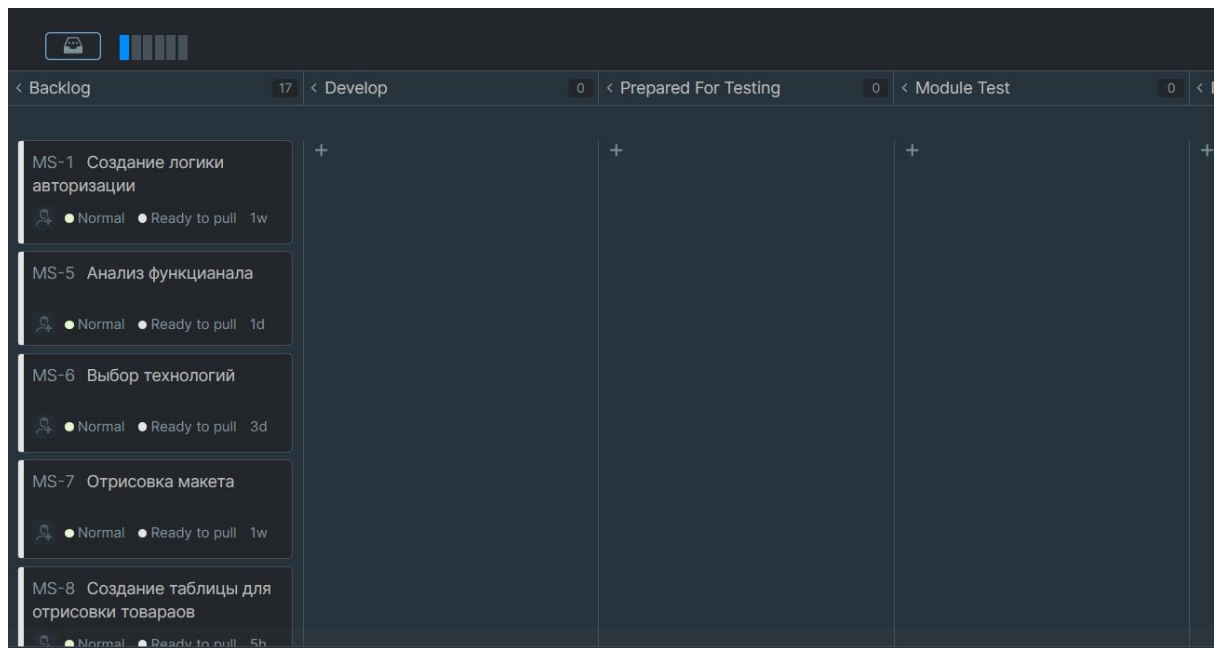


## Этап 5. Настроить столбцы доски для своего проекта

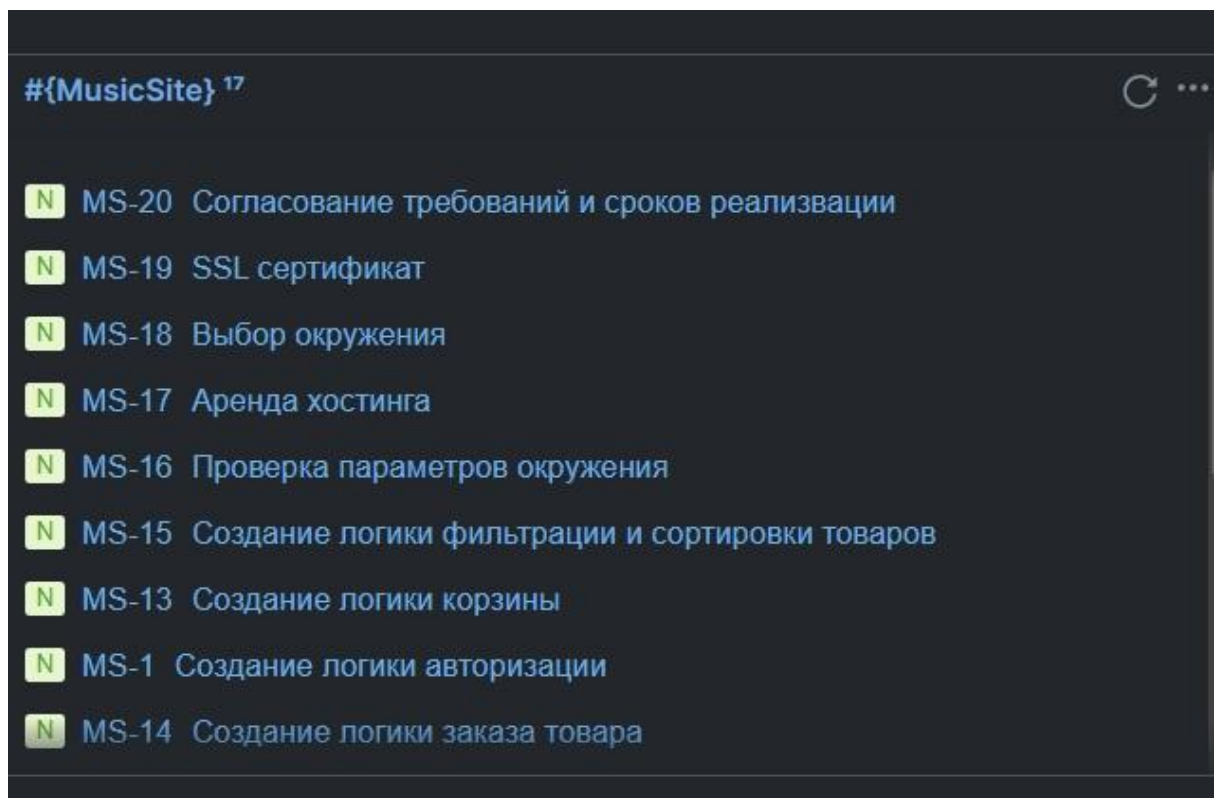


**Этап 6. Создать план работ над проектом и зафиксировать его в YouTrack (создать спринты, релизы и задачи, а также необходимые для работы ветви в репозитории).**

Планировали задачи и ставили задачи команде:




Список задач:



← Created by [liao](#) 1 day ago Updated by [liao](#) 1 day ago

## ★ Аренда хостинга ...

This issue doesn't have a description yet. To add one, click [here](#).

 [Link issue](#)



[liao](#) ● Updated 1 day ago

Estimation: ? → 3d



Write a comment, @mention people

## Этап 7. Симулировать процесс разработки проекта, постепенно закрывая "выполненные" задачи и открывая новые.

Откроем любую задачу чтобы проверить их выполнение и процесс изменения состояния:

← Created by **liao** 2 days ago Updated by **Анна** about 14 hours ago

★ **Фильтр для таблиц интеграции с каталогом** ✎ 📁 ...

🗨 ⌚ ↺ ↻

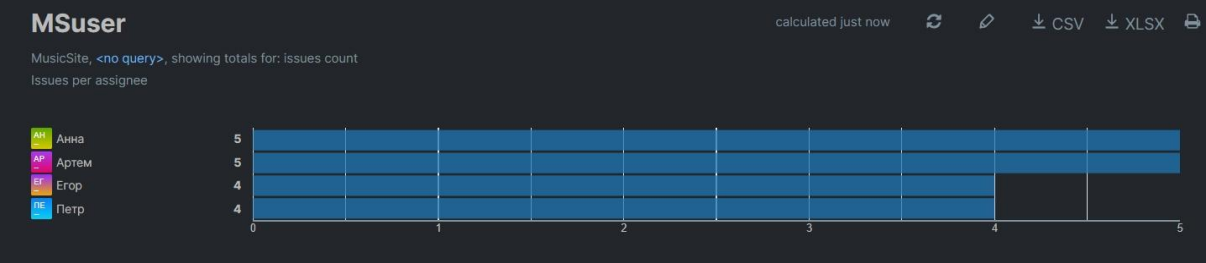
⌚ <b>Анна</b> • Updated about 16 hours ago			
3h 00m	29 Sep 2023	Development	
⌚ <b>liao</b> • Updated about 16 hours ago			
Spent time: ? → 3h			
⌚ <b>Анна</b> • Updated about 16 hours ago			
6h 00m	2 Oct 2023	No work type	
⌚ <b>liao</b> • Updated about 16 hours ago			
Spent time: 3h → 1d 1h			
⌚ <b>Анна</b> • Updated about 16 hours ago			
6h 00m	3 Oct 2023	No work type	
⌚ <b>liao</b> • Updated about 16 hours ago			
Spent time: 1d 1h → 1d 7h			
⌚ <b>Анна</b> • Updated about 16 hours ago			
6h 00m	4 Oct 2023	No work type	

## Этап 8

После завершения снять метрики проекта и предоставить отчет, содержащий описание процесса конфигурации и настройки, описание выбранного workflow, и

сформированные спомощью YouTrack отчеты, отражающие статистику работы над проектом. Обязательнодолжны быть приведены: отчет по исполнителям,burndown-диаграмма, отчет по времени,диаграмма Гантта.

отчет по исполнителям:



отчет по времени:

# MSTimeTable

MusicSite, <no query>, 31 Aug 2023-29 Oct 2023

Time report

per user

per issue

per project

per work item

☐ Show work types

☐ Show remaining/total spent time

Users	Time estimated	Time spent
Total time	534h 00m	579h 00m
<div>АН</div> Анна		145h 00m
<div>АР</div> Артем		149h 00m
<div>ЕГ</div> Егор		143h 00m
<div>ПЕ</div> Петр		142h 00m

# MSTimeTable

calculated 2 minutes

MusicSite, <no query>, 31 Aug 2023-29 Oct 2023

Time report

per user

per issue

per project

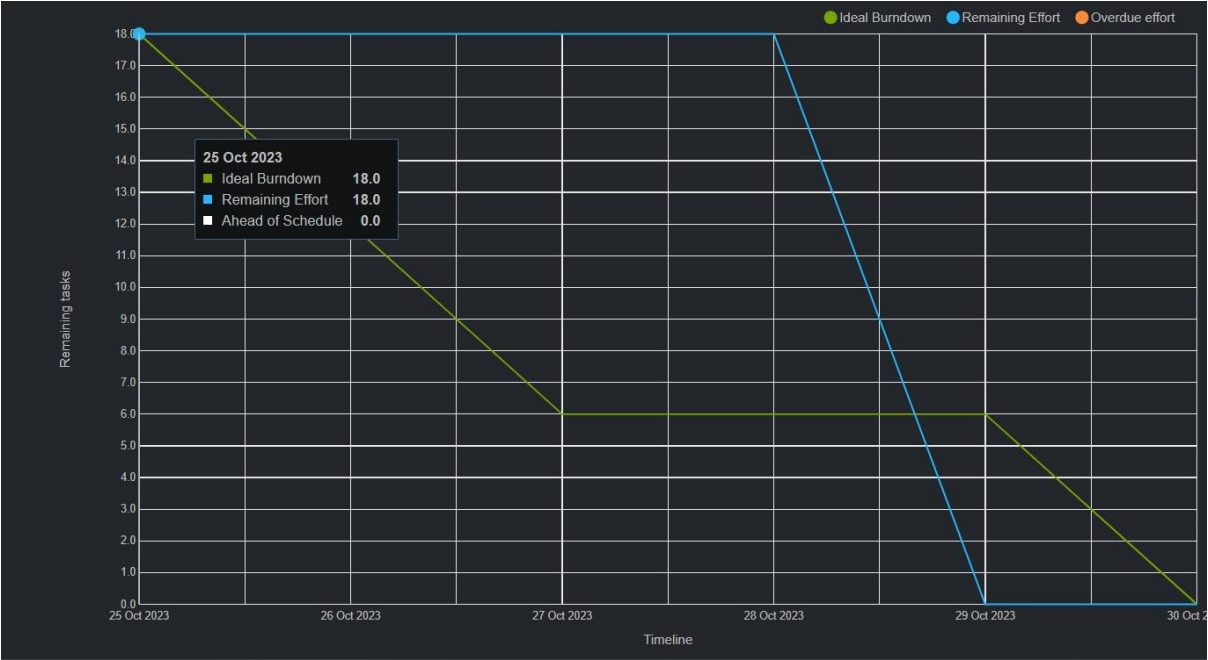
per work item

☐ Show work types

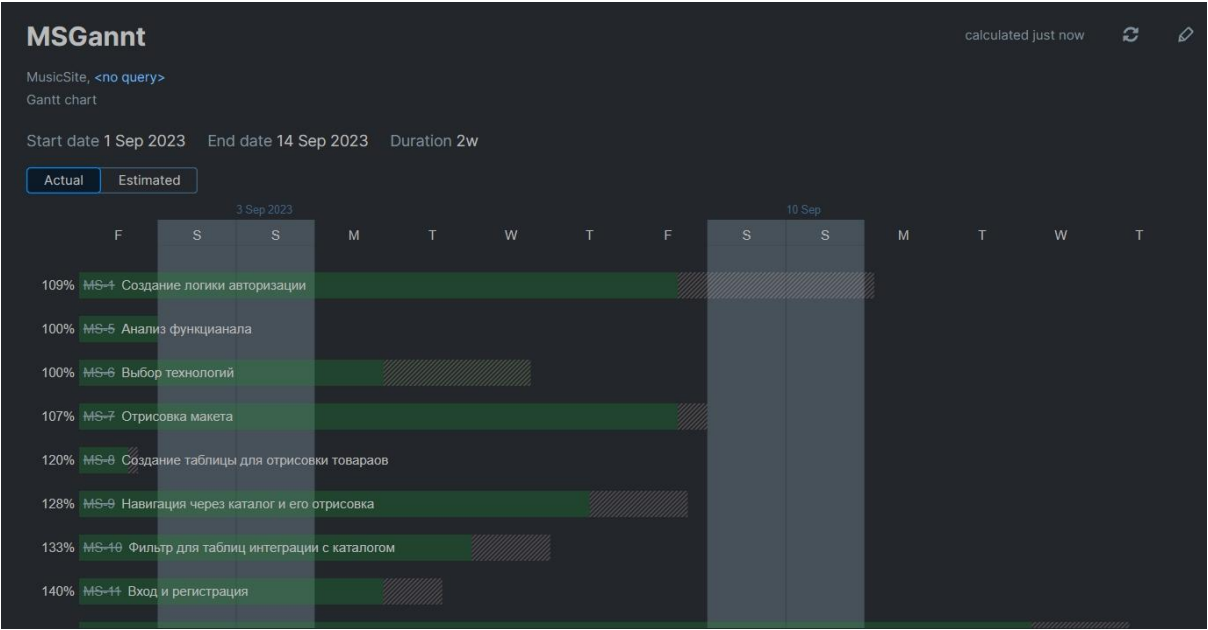
☐ Show remaining/total spent time

Issues	Time estimated	Time spent
Total time	534h 00m	579h 00m
MS-1 Создание логики авторизации	<div></div> 45h 00m	49h 00m
MS-5 Анализ функционала	<div></div> 8h 00m	8h 00m
MS-6 Выбор технологий	<div></div> 30h 00m	15h 00m
MS-7 Отрисовка макета	<div></div> 45h 00m	48h 00m
MS-8 Создание таблицы для отрисовки товараов	<div></div> 5h 00m	6h 00m
MS-9 Навигация через каталог и его отрисовка	<div></div> 36h 00m	46h 00m
MS-10 Фильтр для таблиц интеграции с каталогом	<div></div> 24h 00m	32h 00m
MS-11 Вход и регистрация	<div></div> 15h 00m	21h 00m
MS-12 Создание БД кластера, настройка, создание т...	<div></div> 65h 00m	75h 00m
MS-13 Создание логики корзины	<div></div> 30h 00m	33h 00m

Burndown:



Гантта:





## **Вывод**

В процессе выполнения лабораторной работы мы сконфигурировали YouTrack для работы в команде и выстроили план разработки проекта, при этом интегрировав github репозиторий с нашим YouTrack проектом.

Федеральное государственное автономное  
образовательное учреждение высшего образования

Университет ИТМО

Дисциплина: Экономика программной инженерии

### **Лабораторная работа 3**

Вариант <https://www.muztorg.ru/>

**Выполнили:**

Ляо Ихун

Васильков Александр Сергеевич

**Группа:** Р34131

**Преподаватель:**

Машина Екатерина Алексеевна

2023 г.

Санкт-Петербург

# Задание

Предложить план действий в ситуации, когда прошло 3/4 срока, запланированного на реализацию проекта, а фактически выполнена только половина задач:

1. Определить, какие функции на данный момент еще не завершены и оценить, реализацию каких из них можно отложить для того, чтобы не сдвигать срок выпуска устраивающего заказчика работоспособного продукта с максимально сохраненной функциональностью.
2. Оценить возможность увеличения команды разработчиков для соблюдения сроков проекта, либо попытаться оптимизировать план работ

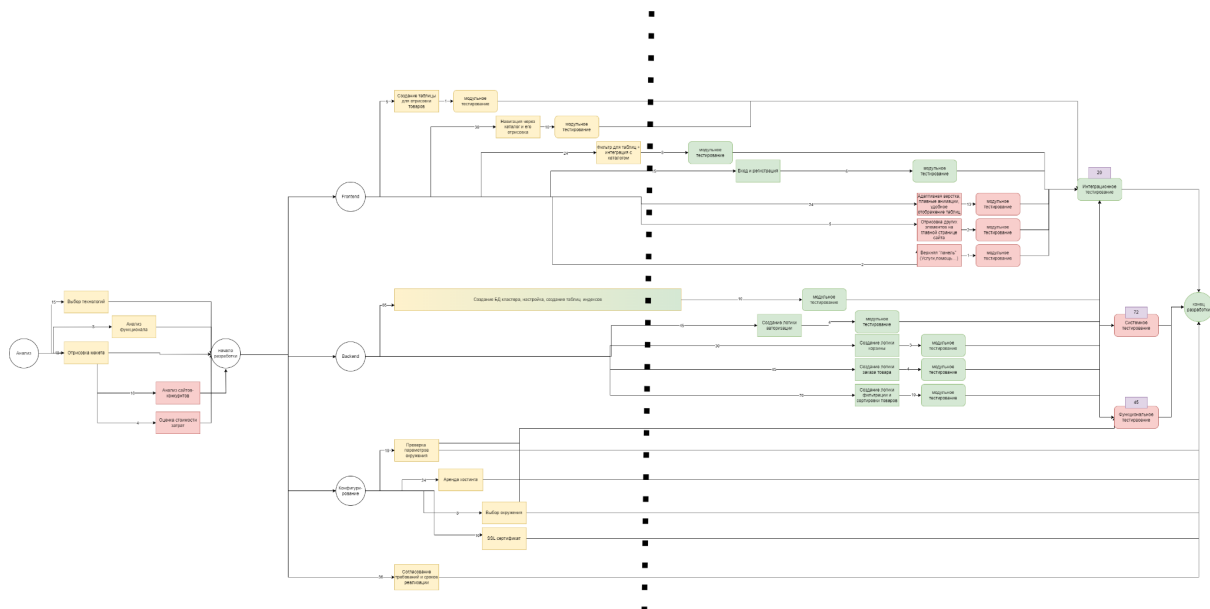
## Выполнение

План действий при задержке

Критический путь

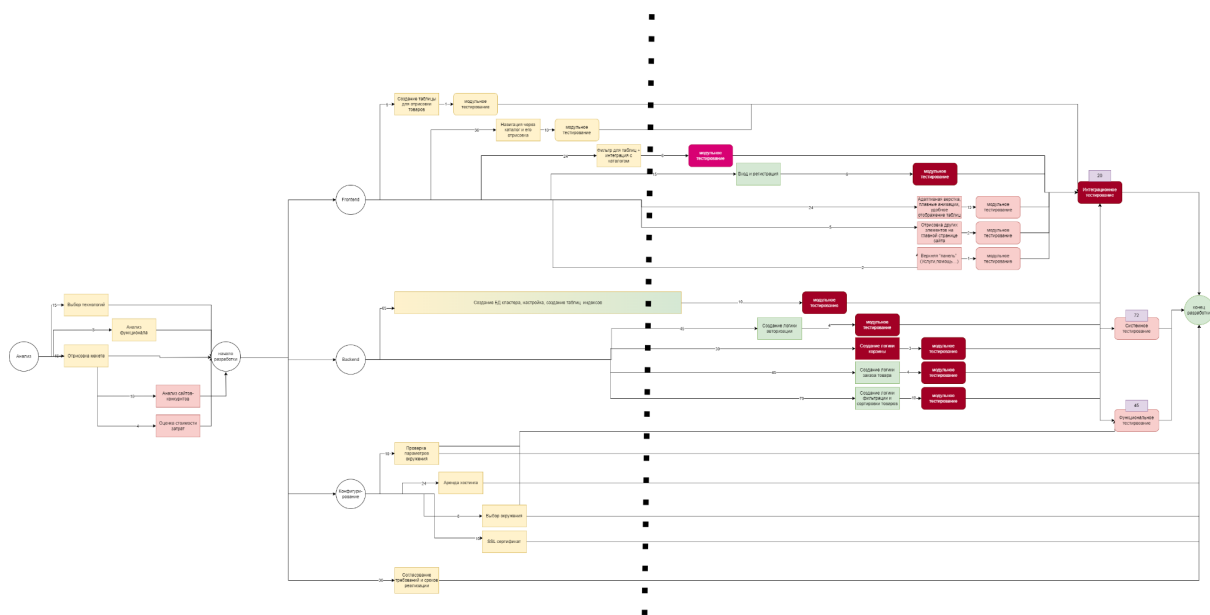
Посмотрим, что входит в **50%** работы, которые мы выполнили за  $\frac{3}{4}$  срока, запланированного на реализацию проекта:  
(Пунктиром показан таймлайн на котором сейчас находится команда!)

- Желтым - отмечены задачи, которые команда успела выполнить за отведенное время.
- Красным(тускло) - задачи, от которых можно отказаться, их реализацию мы отложили еще на этапе определения критического пути.
- Зеленым - задачи, которые осталось сделать за оставшиеся  $\frac{1}{4}$  срока, запланированного на реализацию проекта



Определив сроки, мы можем понять, что реализовать все оставшиеся “зеленые” задачи будет невозможно, поэтому придется от чего-то отказываться.

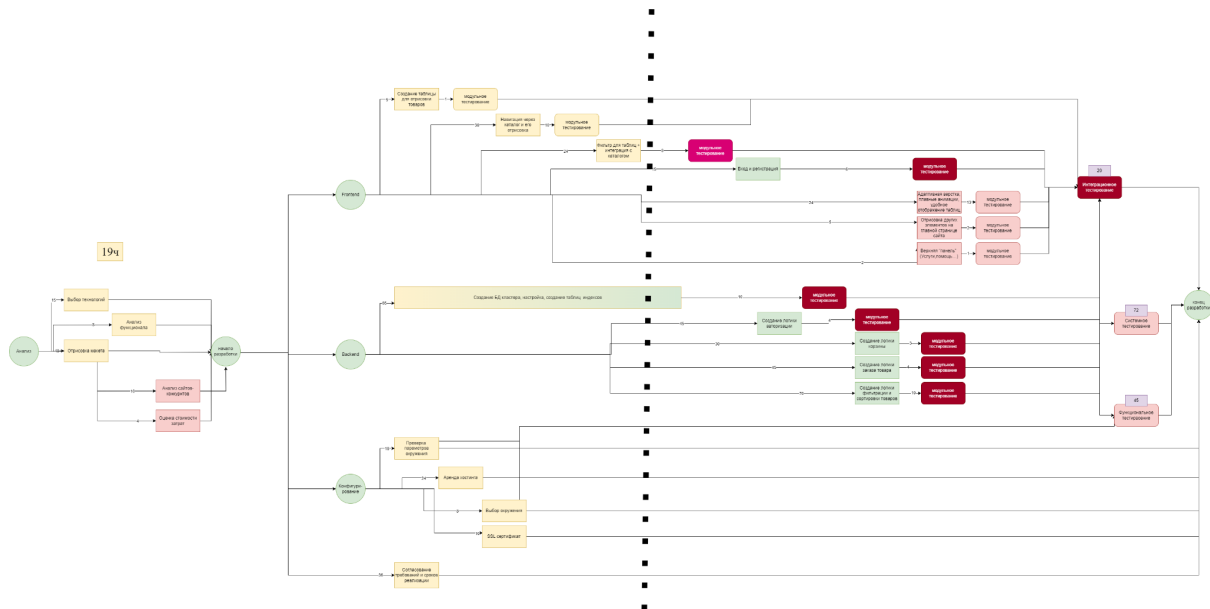
В итоге, чтобы не сдвигать срок выпуска устраивающего заказчика работоспособного продукта с максимально сохраненной функциональностью, мы получаем следующую картину:



Пришлось отказаться от некоторой части модульного тестирования компонентов, интеграционного тестирования и создания логики корзины. Оптимизируя таким образом структуру плана работ команда вполне успеет к релизу, сохранив важный функционал сайта заказчика.

## Оценка возможностей

Оценивая возможность увеличения команды разработчиков для соблюдения сроков проекта, мы пришли к выводу, что наняв еще **одного “супер” разработчика** в нашу команду, мы можем сохранить реализацию логики корзины и предоставить заказчику полноценный(в плане крит пути) функционал.



Если пытаться реализовать все задачи (включая тускло красные), то нужно полностью отказываться от тестирования, добавляя в нашу команду как минимум двух, а лучше трех разработчиков.

## Вывод

В процессе выполнения лабораторной работы, мы представили себя менеджерами в очень плачевной ситуации, когда горят сроки и проект был оценен не совсем корректно, либо команда работала недостаточно слаженно. Мы постарались предпринять возможные меры по перепланированию и оптимизации задач для сохранения сроков сдачи.