ИЗУМРУД СТУДЕНТ

3101675866601

Титульный лист

Направление
- ☐ Естественные науки
- ☐ Инженерные науки
- ☒ Математика и информатика
- ☐ Социальные и гуманитарные науки
- ☐ Экономика и управление

Вариативный блок ☐ 1 ☐ 2 ☒ 3 ☐ 4 ☐ 5

Курс ☐ 1 ☐ 2 ☐ 3 ☒ 4 ☐ 5 ☐ отсутствует

Фамилия АНКУДИНОВА

Имя ПОЛИНА

Отчество АНДРЕЕВНА

Дата рождения 17 04 2004

Город участия ЕКАТЕРИНБУРГ

Аудитория Д3

Дата 02 02 2026 **Подпись**

Пример заполнения

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф

Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я 1 2 3 4 5 6 7 8 9 0

ИЗУМРУД СТУДЕНТ

3101675866601

Проверочный лист

Заполняется участниками

Направление
- ☐ Естественные науки
- ☐ Инженерные науки
- ☒ Математика и информатика
- ☐ Социальные и гуманитарные науки
- ☐ Экономика и управление

Вариативный блок ☐ 1 ☐ 2 ☒ 3 ☐ 4 ☐ 5

Курс ☐ 1 ☐ 2 ☐ 3 ☒ 4 ☐ 5 ☐ отсутствует

Город участия ЕКАТЕРИНБУРГ

Заполняется организаторами

Количество доп листов ☐☐ Количество черновиков к проверке ☐☐

Время выхода с ☐☐ ☐☐ до ☐☐ ☐☐

Протокол проверки
Заполняется жюри

Номер задания	1	2	3	4	5	6	7	8	9	10
Балл члена жюри №1	0		36							
Балл члена жюри №2	0		36							

Итоговый балл 36

Подпись члена жюри №1

Подпись члена жюри №2

Пример заполнения

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф

Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я 1 2 3 4 5 6 7 8 9 0

Блок 3

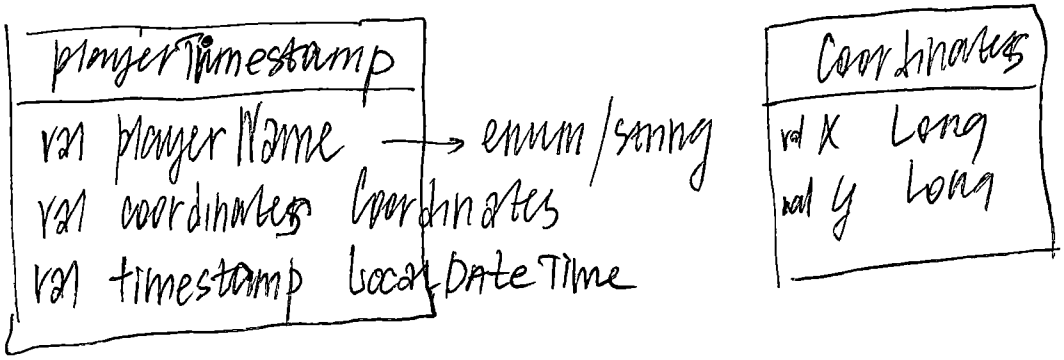
1 Распознавание игроков с 2х камер

1 Делаем снимок поля каждую секунду
На бэкенд отправляем картинку для экономии ресурсов
Три камеры распределяем на три потока

2 На бэкенде у нас размещена ML-модель, уже натренированная на распознавание каждого участника команды

Сделанный снимок футбольного поля обрабатывается на бэкенде → модель распознает: футболиста
его координаты X, Y

После распознавания модель отдает данные серверу в формате data class PlayerTimestamp



1) 155

3 Далее бэкенд должен правильно сохранить полученные от ML-модели данные
Для этого создаем на БД следующую структуру:

отправка на БД будет

players {

„ivan-ivanov" {..}

„andrei_andreev {

„victor-victorovich" {

x_coord Long

y-coord Long

timestamp ~~Timestamp~~ to TimeStamp

}

}

}

хранение на ~~БД~~ будет

и в базе данных

организуем хранение данных.

key	Name	~~timestamp~~ в каком месте был игрок во время T
1	„Ivan Ivanov"	timestamp [X, Y]
2	„Victor-victorovich"	01D02M2S4 01H5S [57, 25]

Благодаря такой структуре мы сможем быстро вычислить скорость, потому что сложность будет O(1) тк массив и обращение по индексу

4 Формируем данные в БД таблице по мере поступления фотографий (раз в 1 сек) и их обработки ML-моделью

---

2 разработать алгоритм замера

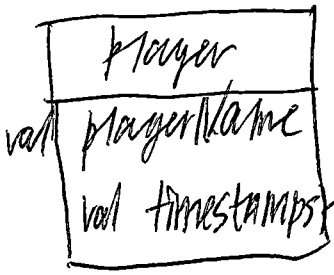
Kotlin-псевдокод

~~средние скорость = players. foreach {~~

```
val players = listOf <Player>
val IvanIvanov = players first()
val сред-скорость_ivan-ivanov = ivan-ivanov. timestamps. map {
    (новое значение списка - предыдущее) • k масштаба в метрах
} . sum() /ivan.ivanov timestamps size()
```

Объяснение алгоритма

~~t0 t1~~ есть получение из БД структура Player,



в каком листе был игрок во время timeStamp

для определения средней скорости игрока
мы находим изменение положения игрока: $x_2 - x_1$, $y_2 - y_1$

и умножаем полученные значения на коэф масштаба k.

полученное новое значение результат местоположения мы округляем и делим на кол-во замеров (размер ~~массива~~ списка timestamps)

Возможность идеи — двигаться по timestamps через интервал 60 секунд (1 минута). 2) 155

~~[illegible]~~

Например для timestamp
01D02M25Y 01H02M03S

нам нужно значение x, y через 60 сек
берем timestamp[01D02M25Y 01H03M03S]
и получаем координаты xy

алгоритм вычисления выносливости

1 Задаем ограничение на повтор 30 раз (30 минут).

2 Берем первое значение, запоминаем x,y
(сами x,y мы берем через timestamps[timestamp первого значения]

3 Переходим ко второму timestamps [timestamp первого знач + 60 sekund]
Теперь тк у нас есть начальная координата (шаг 1) и конечная — вычисляем разницу и умножаем на масштаб k коэффициент

timestamps $[t_1] = x_1; y_1$ координаты t_1

timestamps $[t_1 + 60s] = x_2, y_2$ координаты t_2

$(x_2 - x_1) k$ } пройденное расстояние за 60 сек
$(y_2 - y_1) k$ } поминутное распределение скорости

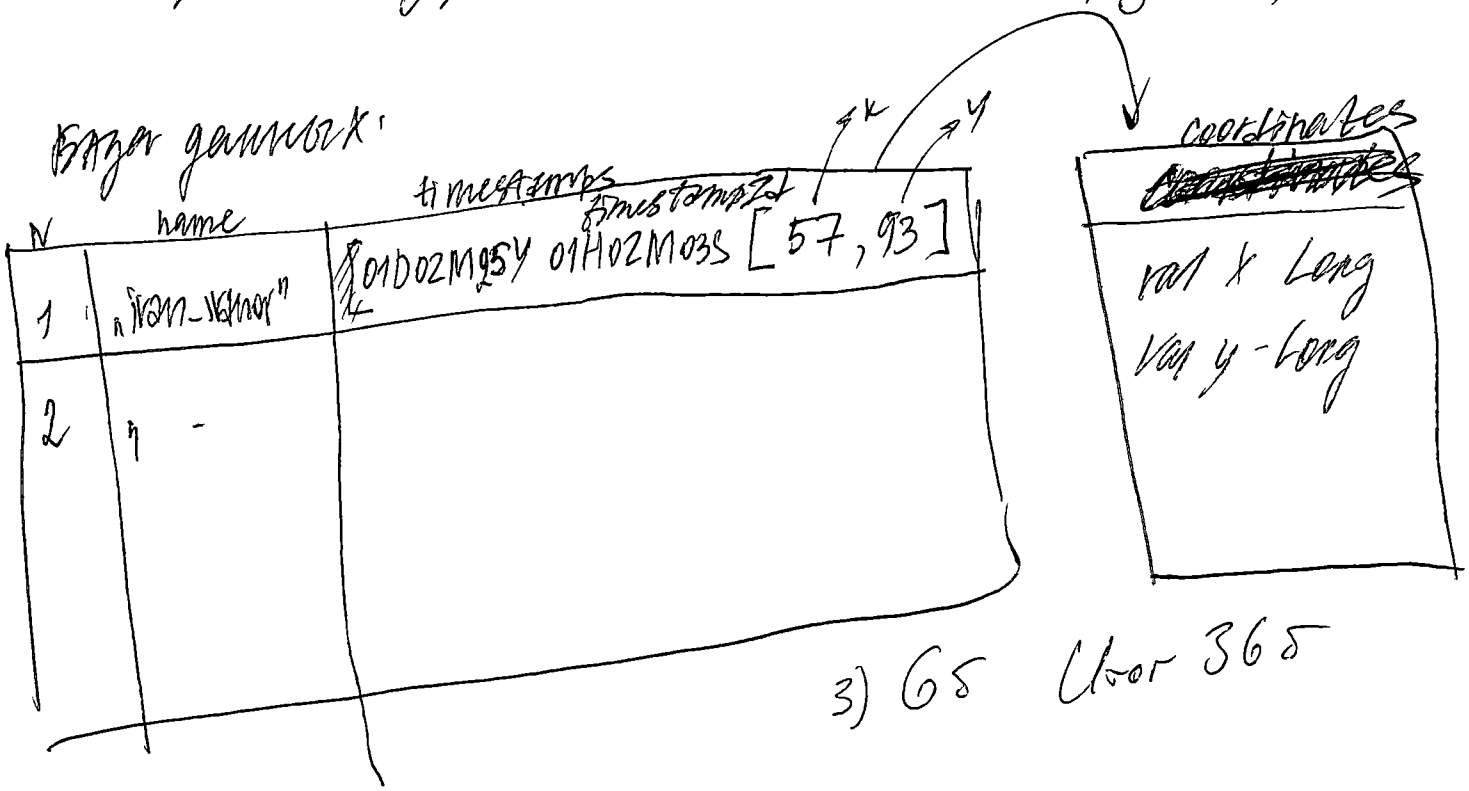
вычисленные значения сохраняем в массиве поминутное_распределение_скорости: [illegible]

заканчиваем проход после 30 раз (30 минут условие)

Итог: получим массив поминутное_распределение_скорости, в котором лежат 30 значений в м/с

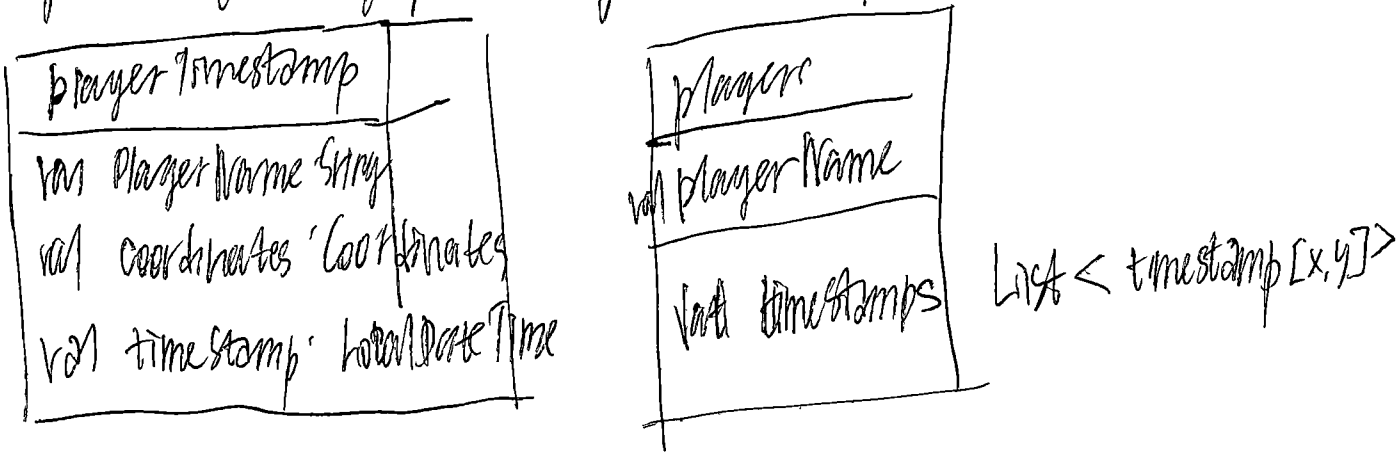
В итоге формируем среднюю скорость и выносливость по каждому игроку при необходимости можно отправить данные на клиент (например android app „Здоровье футболиста")

3 Архитектура



3) Об Coor 365

ML Модель модель возвращает данные в формате



mapping DTO в Модельные данные

Вторая данных в системе была описана ранее