Описание изменений для ИИ (AI Replication Instructions)

Вам необходимо внести следующие изменения в проект, чтобы реализовать экран «Отчет о завершении тренировки» для роли `РУКОВОДИТЕЛЬ`. Внимательно следуйте шагам:

Шаг 1: Обновление SimulationBoard.tsx

Откройте файл `frontend/src/widgets/SimulationBoard/SimulationBoard.tsx`. Добавьте поддержку свойства `isReadOnly` для компонента:

1. Найдите интерфейс `SimulationBoardProps` и добавьте в него `isReadOnly?: boolean;`.

```
interface SimulationBoardProps {
  activeRole?: string;
  isReadOnly?: boolean;
}
```

2. Обновите объявление компонента:

```
export const SimulationBoard: React.FC<SimulationBoardProps> = ({ activeRole,
isReadOnly }) => {
```

3. Найдите все места в компоненте, где рендерятся панели управления или элементы пользовательского интерфейса поверх карты, и скройте их, если `isReadOnly` истинно `!isReadOnly && (...)`. (Ориентировочные изменения - закрыть панель управления `RadioConsole` и кнопки масштаба таким условием).
4. Запретите клики по канвасу и показ хинтов, если включен режим чтения, или оберните панель управления в условный рендер: `{!isReadOnly && <div ...> controls </div>}`

Шаг 2: Экспорт функции в DispatcherSidebar.tsx

Откройте файл `frontend/src/features/DispatcherControls/DispatcherSidebar.tsx`. Найдите объявление константы `parseDispatcherJournal` (около строки 212) и добавьте ключевое слово `export`, чтобы функцию можно было использовать в другом файле:

```
export const parseDispatcherJournal = (
  snapshotData: Record<string, unknown> | null | undefined,
): DispatcherJournalEntry[] => {
...
```

Шаг 3: Создание TrainingCompletionReport.tsx

Создайте **НОВЫЙ** файл по пути `frontend/src/features/TrainingLeadControls/TrainingCompletionReport.tsx` и вставьте в него следующий код:

```tsx
import React, { useMemo } from 'react';
import type { SessionStateBundleDto } from '../../shared/api/types';
import { PixelButton } from '../../shared/ui/PixelButton';
import { parseDispatcherJournal } from '../DispatcherControls/DispatcherSidebar';
import { SimulationBoard } from '../../widgets/SimulationBoard/SimulationBoard';

type TrainingCompletionReportProps = {
    bundle: SessionStateBundleDto | null;
    onClose?: () => void;
};

export const TrainingCompletionReport: React.FC<TrainingCompletionReportProps> = ({
bundle, onClose }) => {
    const journalEntries = useMemo(() => {
        return parseDispatcherJournal(bundle?.snapshot?.snapshot_data);
    }, [bundle?.snapshot?.snapshot_data]);

    return (
        <div className="flex flex-col h-full bg-[#111] overflow-hidden text-gray-200
p-4">
            <div className="flex items-center justify-between mb-4 border-b-2
border-gray-700 pb-2 flex-shrink-0">
                <h2 className="text-sm text-green-400 font-bold uppercase tracking-
wider">
                    ОТЧЕТ О ЗАВЕРШЕНИИ ТРЕНИРОВКИ
                </h2>
                {onClose && (
                    <PixelButton size="sm" variant="default" onClick={onClose}>
                        ЗАКРЫТЬ
                    </PixelButton>
                )}
            </div>

            <div className="flex-1 flex flex-col min-h-0 gap-4">
                {/* UPPER HALF: Journal */}
                <div className="flex-1 flex flex-col min-h-0 border-2 border-gray-
800 bg-[#161616] rounded-sm p-3">
                    <h3 className="text-xs text-blue-300 uppercase mb-2">Журнал
диспетчера</h3>
                    <div className="flex-1 overflow-y-auto space-y-2 pr-2 custom-
scrollbar">
                        {journalEntries.length === 0 ? (
                            <div className="text-[10px] text-gray-500 italic py-4
text-center">Журнал пуст</div>
                        ) : (
                            journalEntries.map((entry: any) => (
                                <div key={entry.id} className="text-[9px] border-b
border-gray-800 pb-2">
                                    <div className="flex justify-between text-gray-
400 mb-1">
                                        <span>{new
```

```
Date(entry.created_at).toLocaleTimeString('ru-RU')}</span>
                                        <span>{entry.author}</span>
                                    </div>
                                    <div className="text-gray-200">{entry.text}
</div>
                                </div>
                            ))
                        )}
                    </div>
                </div>

                {/* LOWER HALF: Schemas */}
                <div className="flex-1 min-h-0 grid grid-cols-3 gap-4">
                    <div className="flex flex-col h-full border-2 border-gray-800
bg-[#161616] rounded-sm p-2">
                        <h3 className="text-[10px] text-blue-300 uppercase mb-2
text-center">Схема: ШТАБ</h3>
                        <div className="flex-1 border border-black overflow-hidden
relative">
                            <SimulationBoard activeRole="ШТАБ" isReadOnly />
                        </div>
                    </div>
                    <div className="flex flex-col h-full border-2 border-gray-800
bg-[#161616] rounded-sm p-2">
                        <h3 className="text-[10px] text-blue-300 uppercase mb-2
text-center">Схема: БУ-1</h3>
                        <div className="flex-1 border border-black overflow-hidden
relative">
                            <SimulationBoard activeRole="БУ - 1" isReadOnly />
                        </div>
                    </div>
                    <div className="flex flex-col h-full border-2 border-gray-800
bg-[#161616] rounded-sm p-2">
                        <h3 className="text-[10px] text-blue-300 uppercase mb-2
text-center">Схема: БУ-2</h3>
                        <div className="flex-1 border border-black overflow-hidden
relative">
                            <SimulationBoard activeRole="БУ - 2" isReadOnly />
                        </div>
                    </div>
                </div>
            </div>
        </div>
    );
};
```

Шаг 4: Интеграция в TrainingLeadWorkspace.tsx

Откройте `frontend/src/features/TrainingLeadControls/TrainingLeadWorkspace.tsx`.

1. Добавьте импорт нового компонента рядом с другими локальными импортами:

```
import { TrainingCompletionReport } from './TrainingCompletionReport';
```

2. Найдите блок, где вычисляется состояние `isLessonCompleted` или создайте его внутри функции компонента:

```
  const isLessonCompleted =
    realtimeBundle?.session.status === 'COMPLETED' ||
    realtimeBundle?.lesson_state?.status === 'COMPLETED';
```

3. Найдите главный `return` компонента. Прямо перед ним вставьте ранний возврат:

```
  if (isLessonCompleted) {
    return <TrainingCompletionReport bundle={realtimeBundle} />;
  }

  return (
    <div className="flex-1 flex overflow-hidden">
    // ... остальной код
```

Эти четыре шага полностью повторяют все изменения, сделанные в предыдущей сессии.