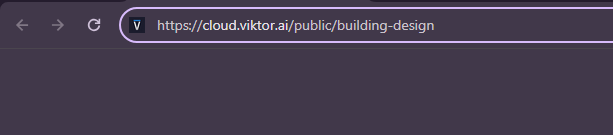
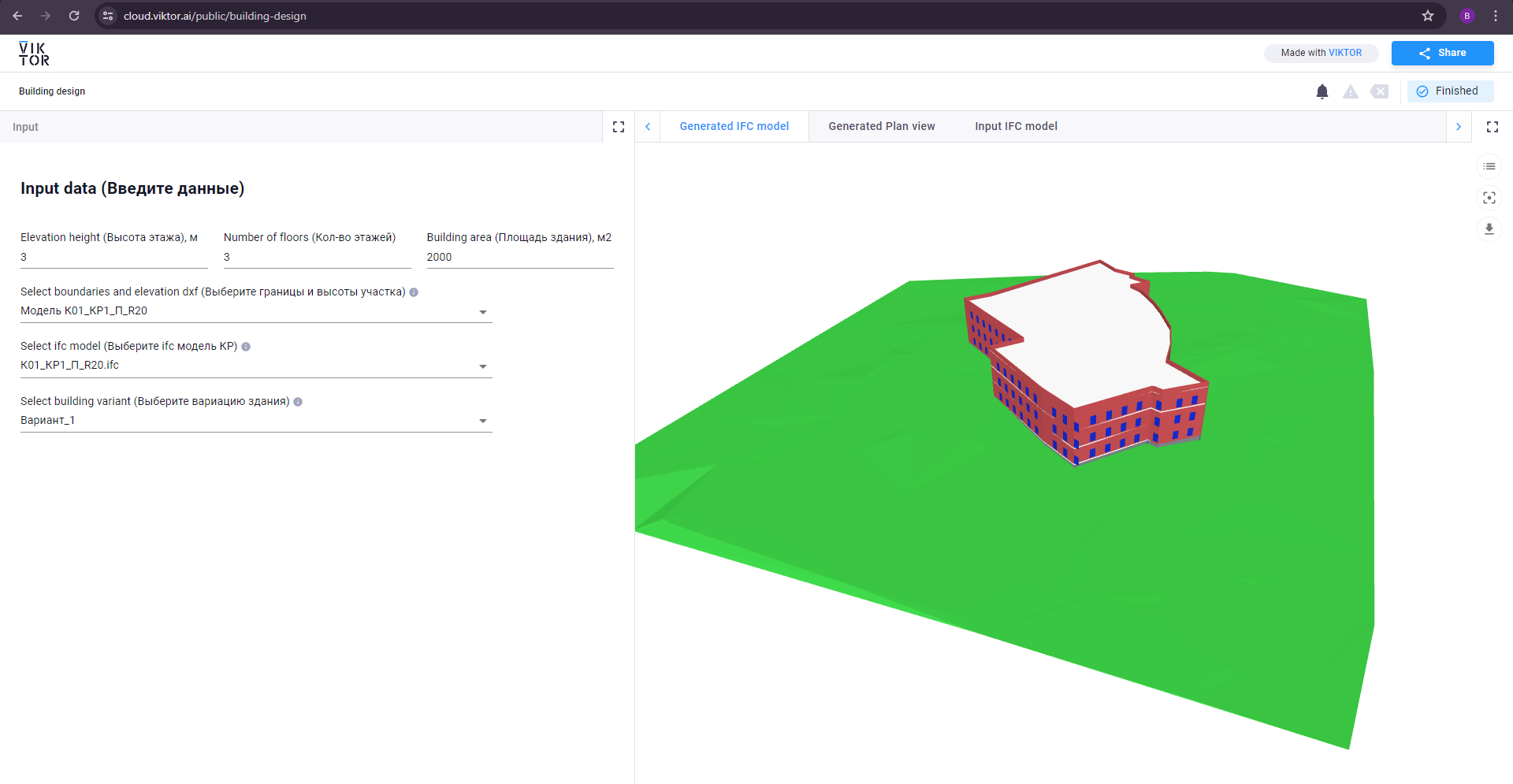
Инструкция использования программы

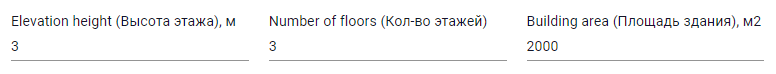
1. Перейти по ссылке <https://cloud.viktor.ai/public/building-design>



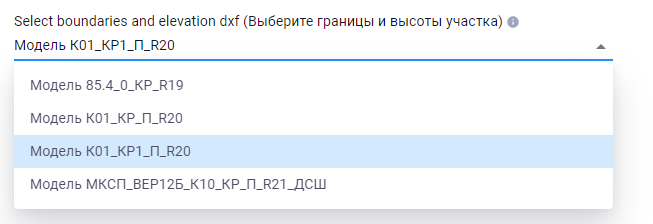
1. Подождать пока сгенерируется здание по предварительно заданным данным.



1. При необходимости изменить данные высоты этажа, количества этажей и площади здания.

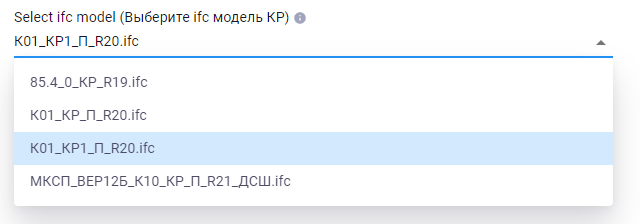


1. При необходимости изменить тестовую выборку пары границ участка и его высот которые были даны к указанным в выпадающем списке моделям.

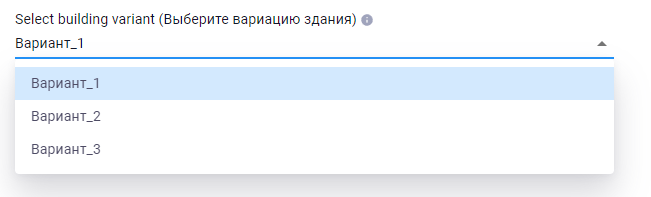


Примечание: Бесплатное использование веб-платформы VICTOR не позволяет пользователю загружать файлы. Поэтому файлы тестовой выборки были заранее загружены на платформу (кроме 5-го, т.к. модель слишком большая) и предоставлена возможность выбора из выпадающего списка.

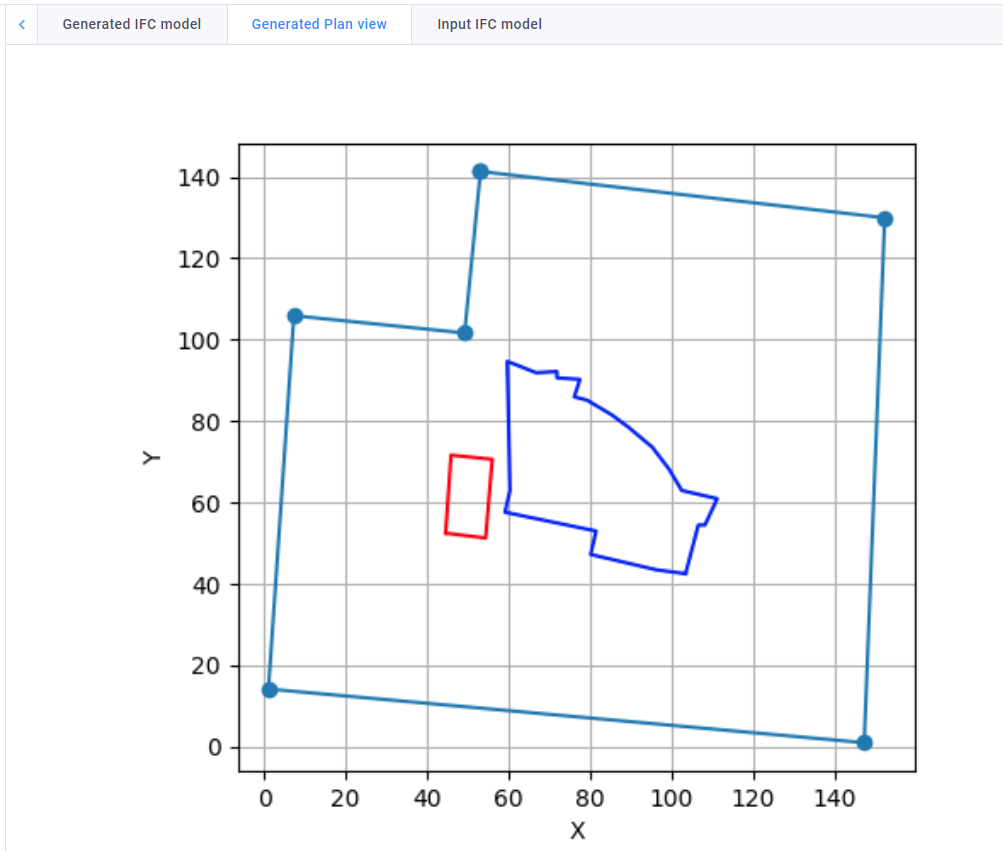
1. При необходимости изменить тестовую выборку предоставленных IFC моделей.



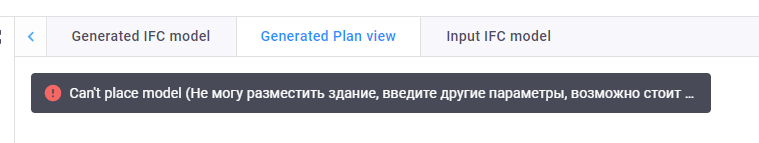
1. При необходимости выбрать один из трех вариантов предложенных моделей.



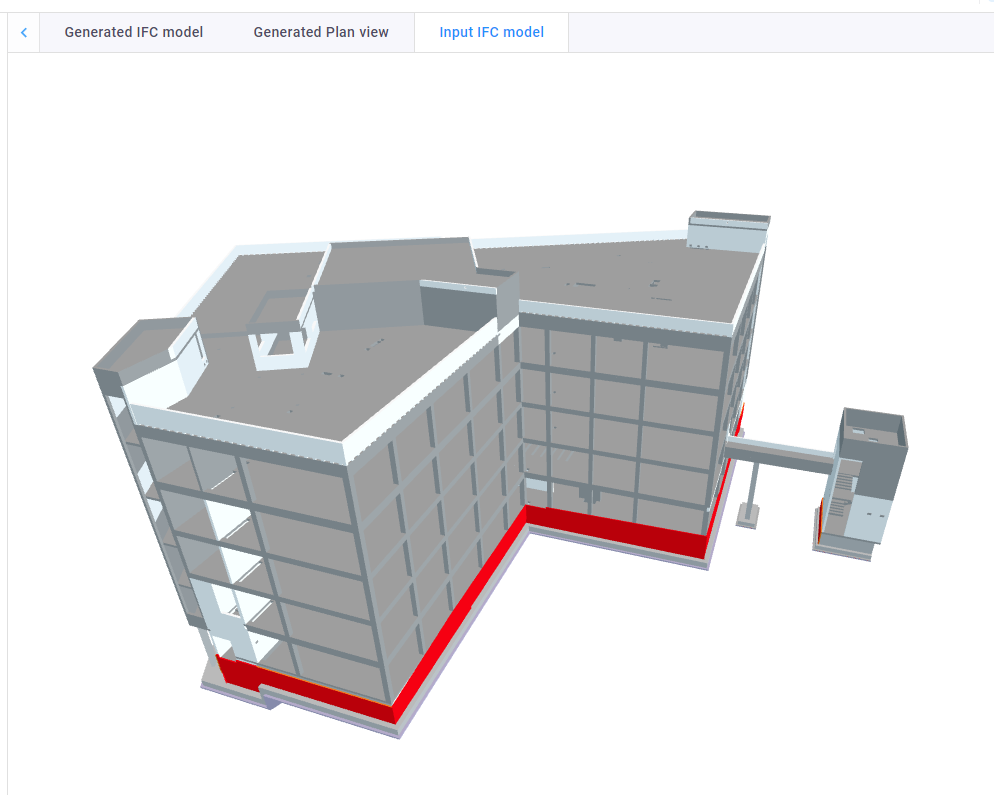
1. На вкладке *"Generated Plan view"* можно увидеть схему границ участка с размещенным на нем сгенерированным зданием (красный квадрат – это внутреннее ограничение площадки, было в нескольких файлах).



1. При возникновении ошибки *"Can't place model (Не могу разместить здание, введите другие параметры, возможно стоит уменьшить площадь здания)"* необходимо внести другие данные, например уменьшить площадь здания, т.к. размещение здания в указанных габаритах невозможно.



1. На вкладке *"Input IFC model"* можно увидеть исходную загруженную модель из тестовой выборки.



Алгоритмы

# Алгоритм извлечения плана здания из загруженной модели IFC:

1. С помощью библиотеки ifcopenshell открываем файл модели.

2. Начиная с цокольного этажа берем уровни пока не найдем тот на котором есть стены типов "IfcWall" или "IfcWallStandardCase".

3. Достаем из найденных стен их координаты начала и конца.

4. Формируем полигон контура здания из полученных линий стен.

# Алгоритм извлечения высот:

1. Извлекаем все данные о всех объектах из файла DXF с помощью функции extract\_all\_entities\_data.

2. Находим все текстовые объекты (TEXT), содержащие числовые значения высот (соответствующие регулярному выражению ^\d+\.\d+$).

3. Для каждого найденного текстового объекта с высотой находим ближайший объект типа INSERT с помощью функции find\_close\_vec.

4. Формирует список значений высот и список координат, соответствующих этим высотам.

5. Возвращает словарь с данными о высотах.

# Алгоритм поиска подобных объектов.

1. Готовим базу объектов, среди которых будут искаться подобные.

Для этого приводим координаты объектов из базы к единому масштабу, зажимая в диапазон [1, 2].

Отрисовываем внешний контур каждого объекта и трансформируем его в растровое изображение.

При помощи сверточной сети ResNet50 с отсеченной регрессионной головой формируем скрытое представление каждого объекта в виде вектор-строки длины 2048.

Сохраняем координаты объектов в формате {имя: [[массив\_х], [массив\_y]]}

Вектора признаков сохраняем в виде kd-дерева для быстрого поиска подобных.

2. Получаем координаты вершин объекта, подобные которому мы должны найти.

3. Масштабируем координаты объекта, трансформируем его в изображение и вытягиваем признаки.

4. В kd-дереве ищем k подобных векторов. По словарю возвращаем их координаты для дальнейшего преобразования в объекты для размещения.

# Алгоритм расположения объектов:

1. Получаем DXF файл с габаритами площадки.

2. Вытягиваем из него объекты LWPOLYLINE с помощью библиотеки ezdxf. Если объект один, значит внутренних радиусов нет. В противном случае у площадки есть «отверстия», границы которых нельзя пересекать.

3. Нормализуем координаты вершин площадки, располагая их в первой четверти плоскости и смещая к отметке (0, 0). Вектор смещения также используется для перемещения отметок высот.

4. Получаем координаты вершин объекта, который нужно разместить, и масштабируем его до площади, заданной пользователем.

6. Смещаем объект таким образом, чтобы его центроид совпал с центроидом площадки.

7. Начинаем итеративный процесс замощения. Формируем расходящуюся из центра спираль точек. Последовательно двигаемся из центра наружу, смещая центр объекта на точки и поворачивая объект на каждой точке 360 раз с углом в 1 градус, проверяя объекты на полное вхождение.

8. Если решение найдено – возвращаем координаты объекта. В противном случае выкидываем ошибку и просим пользователя изменить входные данные или параметры.

# Алгоритм генерации модели IFC:

1. С помощью библиотеки ifcopenshell создаем файл модели.

2. По полученным координатам стен строим стены с окнами.

3. По полученным координатам и отметкам высот строим поверхность земли на участке с триангуляцией.

4. При необходимости меняется высота этажа и их количество.