Quick Start

```
pnpm install # or npm install
pnpm dev # or npm run dev
```

虚拟看房项目技术选型

PC端

本虚拟看房项目PC端使用`nextjs` + `threejs`构建

技术优势

用户相对易用

作为一款面向C端软件, 我们首要考虑的就是客户的体验. 由于使用unity框架构建出来的产物过于庞大, 可移植性低, 并且需要用户额外下载对应的软件, 所以我们使用传统的Web框架. 其构建出来的产物可移植性高, 灵活度高.

对开发人员要求低/成本低

目前市面上使用unity的开发人员较少, 薪资相对较高. Web开发人员比较好找, 薪资也相对较低. 使用Web开发人员进行开发可以减少企业的大量成本.

可扩展性好/代码复用性高

将模型和底层代码打包后可以很大程度在多个平台复用(Web页面/原生安卓,IOS(RN)/H5小程序(uni-app等框架)/桌面端程序(Electron)), 可以节省开发人员的大量时间和企业成本.

容易与现有项目对接

本项目为产物为一个Web页面, 开发完成后不仅可以作为一个子路由嵌入房地产公司的官网中, 还可以作为一个iFrame嵌入其他大型平台.

VR场景构建

场景构建

由于用代码直接构建三维场景是不现实也是不合理的, 所以本项目使用C4D作为场景构建的主要工具, 代码层面主要是控制摄像机的各种活动. `threejs`提供了可以导入3D模型`*.obj, */mtl`的接口, 开发人员可以直接和建模对接, 完成相应的工作.

VR内容

`threejs`中有直接创建VR内容的接口, 详情可查看https://threejs.org/docs/#manual/en/introduction/How-to-create-VR-content

核心代码

- 加载模型(实际业务场景中应该为网络请求)

```
const loadResource = () => {
  return new Promise((resolve, reject) => {
    const objLoader = new OBJLoader();
    const mtlLoader = new MTLLoader();
    mtlLoader.load('/models/source/scence.mtl', mtl => {
      objLoader.setMaterials(mtl);
      objLoader.load('/models/source/scence.obj', res => {
        resolve(res);
      }, undefined, reject);
    }, undefined, reject);
  });
};
```

- 创建场景(加载模型以及摄像机)

```
const start = async (ref) => {
  const scene = new THREE.Scene();
  const camera = new THREE.PerspectiveCamera(75, window.innerWidth /
window.innerHeight, 0.1, 1000);
  const renderer = new THREE.WebGLRenderer();
  renderer.setSize(window.innerWidth, window.innerHeight);
  ref.appendChild(renderer.domElement);

  models.forEach(model => scene.add(model));
  const mainScene = await loadResource();
  scene.add(mainScene);
  camera.position.x = 0;
  camera.position.z = 3;
  camera.position.y = 0;
  const controls = new OrbitControls(camera, renderer.domElement);
  controls.addEventListener('change', render);
  controls.minDistance = 2;
  controls.maxDistance = 50;
  controls.enablePan = false;
  function render() {
    renderer.render(scene, camera);
  }
  render();
};
```