开发一个名为 "everauadmin" 的全栈应用，并结合 Django、Next.js 等技术工具来实现你提到的各个功能，听起来是一个非常有趣的项目。我会一步一步地非常详细地为你写出整个开发过程，并确保你能在 VS Code 中无缝进行开发。这个项目的核心模块包括：

1. 定时截取桌面截图。
2. 获取本机的序列号。
3. 集成 RustDesk 的桌面共享功能。
4. 当员工登录 LightSpeed 并进入结账页面时，提醒他们使用 Deputy 打卡。

为了方便你跟进，我会从项目的环境搭建开始，到功能实现，每一步都详细说明。

**第1步：设置开发环境**

**1.1 安装必要工具**

* 安装 [Python](https://www.python.org/downloads/)（Django 后端需要）和 [Node.js](https://nodejs.org/)（Next.js 前端需要）。
* 安装 [VS Code](https://code.visualstudio.com/) 作为代码编辑器。
* 安装 Git 进行版本控制（可选）。

**1.2 创建 Django 项目**

1. 在终端中，创建虚拟环境并激活它：

bash

复制代码

python -m venv env

source env/bin/activate # Linux 或 Mac

env\Scripts\activate # Windows

1. 安装 Django：

bash

复制代码

pip install django

1. 创建 Django 项目并启动 Django 应用：

bash

复制代码

django-admin startproject everauadmin

cd everauadmin

django-admin startapp core

1. 配置数据库（使用 SQLite 或 PostgreSQL，根据需要）： 在 settings.py 中配置数据库设置：

python

复制代码

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE\_DIR / 'db.sqlite3',

}

}

1. 迁移数据库：

bash

复制代码

python manage.py migrate

1. 创建超级用户以访问 Django Admin：

bash

复制代码

python manage.py createsuperuser

1. 运行 Django 开发服务器，检查是否成功：

bash

复制代码

python manage.py runserver

现在 Django 项目已经搭建成功，接下来我们会搭建 Next.js 前端。

**第2步：设置 Next.js 前端**

**2.1 初始化 Next.js 项目**

1. 在项目根目录下创建 frontend 文件夹，并在该文件夹中运行以下命令初始化 Next.js 项目：

bash

复制代码

npx create-next-app frontend

cd frontend

npm install

1. 修改 package.json 文件，确保前后端在一个命令下同时启动。添加以下脚本：

json

复制代码

"scripts": {

"dev": "concurrently \"npm run dev --prefix frontend\" \"python manage.py runserver\""

}

同时安装 concurrently，用于同时运行多个命令：

bash

复制代码

npm install concurrently

1. 现在你可以通过以下命令同时运行前后端：

bash

复制代码

npm run dev

**2.2 配置 Django 和 Next.js 的交互**

为了让前后端能进行数据通信，我们需要配置 Django 作为 API 后端，Next.js 作为前端。使用 Django 的 django-rest-framework 来创建 API。

1. 安装 django-rest-framework：

bash

复制代码

pip install djangorestframework

1. 在 settings.py 中添加：

python

复制代码

INSTALLED\_APPS = [

...

'rest\_framework',

]

1. 创建 API 视图： 在 core/views.py 中添加获取本机序列号的 API：

python

复制代码

from rest\_framework.decorators import api\_view

from rest\_framework.response import Response

import subprocess

@api\_view(['GET'])

def get\_serial\_number(request):

try:

serial\_number = subprocess.check\_output('wmic bios get serialnumber', shell=True).decode().split('\n')[1].strip()

return Response({'serial\_number': serial\_number})

except Exception as e:

return Response({'error': str(e)}, status=500)

1. 在 core/urls.py 中定义路由：

python

复制代码

from django.urls import path

from . import views

urlpatterns = [

path('get-serial/', views.get\_serial\_number),

]

1. 在 everauadmin/urls.py 中添加 API 路由：

python

复制代码

from django.urls import path, include

urlpatterns = [

path('api/', include('core.urls')),

]

**2.3 Next.js 前端调用 API**

1. 在 frontend/pages/index.js 中编写前端调用 Django API 的逻辑：

javascript

复制代码

import { useState, useEffect } from 'react';

export default function Home() {

const [serialNumber, setSerialNumber] = useState('');

useEffect(() => {

fetch('http://localhost:8000/api/get-serial/')

.then(response => response.json())

.then(data => setSerialNumber(data.serial\_number || 'Error fetching serial number'));

}, []);

return (

<div>

<h1>本机序列号: {serialNumber}</h1>

</div>

);

}

**第3步：定时安静截图桌面**

接下来，我们将使用 Python 的 pyautogui 库来实现定时截取桌面截图功能。

**3.1 安装 pyautogui：**

bash

复制代码

pip install pyautogui

**3.2 编写截图功能**

在 core/views.py 中编写一个新的 API 来定时截取桌面截图：

python

复制代码

import pyautogui

from datetime import datetime

@api\_view(['GET'])

def take\_screenshot(request):

try:

now = datetime.now().strftime('%Y%m%d%H%M%S')

screenshot = pyautogui.screenshot()

file\_path = f'screenshots/screenshot\_{now}.png'

screenshot.save(file\_path)

return Response({'message': 'Screenshot saved', 'path': file\_path})

except Exception as e:

return Response({'error': str(e)}, status=500)

**3.3 添加路由**

在 core/urls.py 中添加截图 API：

python

复制代码

urlpatterns += [

path('take-screenshot/', views.take\_screenshot),

]

**第4步：集成 RustDesk 和 LightSpeed 功能**

**4.1 集成 RustDesk**

RustDesk 的桌面共享可以通过其 API 或 SDK 实现。你可以在前端页面嵌入 RustDesk 客户端或者调用其 API。

**4.2 LightSpeed 和 Deputy 打卡提醒**

对于 LightSpeed，你可以在 Next.js 前端中检测登录和结账页面状态，调用 Deputy 的 API 或者通过前端弹出提醒。

**总结**

我已经为你详细描述了项目的每一个步骤，包括 Django 后端、Next.js 前端、定时截图、获取本机序列号等。接下来你可以根据需要继续开发每个功能模块。