

سیستم ساده انتخاب واحد دانشجو

➤ توضیح کلی پروژه:

این سامانه یک وب‌اپلیکیشن است که امکان مدیریت اطلاعات دانشجویان را فراهم می‌کند. کاربران می‌توانند از طریق رابط کاربری (فرانت‌اند) اطلاعات دانشجویان، اساتید و دروس را مدیریت کرده و از طریق API های بک‌اند، داده‌ها را در پایگاه داده SQLite ذخیره و بازیابی کنند. این پروژه با استفاده از FastAPI برای بک‌اند، HTML, CSS, Js برای فرانت‌اند و Nginx به عنوان وب‌سرور پیاده‌سازی شده است.

➤ ویژگی‌ها:

مدیریت دانشجویان: ثبت، ویرایش، حذف و نمایش اطلاعات دانشجویان شامل شماره دانشجویی، نام، کد ملی، تاریخ تولد، شهر تولد، آدرس، کد پستی، تلفن، نام پدر، سریال شناسنامه، دانشکده، رشته، وضعیت تاهل و کد درس.

مدیریت اساتید: ثبت، ویرایش، حذف و نمایش اطلاعات اساتید شامل کد استاد، نام، کد ملی، تاریخ تولد، شهر تولد، آدرس، کد پستی، تلفن، دانشکده و رشته.

مدیریت دروس: ثبت، ویرایش، حذف و نمایش اطلاعات دروس شامل کد درس، نام درس، دانشکده و تعداد واحد.

اعتبارسنجی داده‌ها: اعتبارسنجی دقیق و سخت‌گیرانه داده‌های ورودی مانند کد ملی، شماره دانشجویی، تلفن، تاریخ تولد در بک‌اند و فرانت‌اند.

کانتینرسازی پیشرفته: استفاده از Docker و Docker Compose برای اجرای هماهنگ بک‌اند و فرانت‌اند.

➤ فناوری‌ها:

بک‌اند: Python 3.13, FastAPI, SQLAlchemy, Pydantic

پایگاه داده: SQLite

فرانت‌اند: HTML, CSS, JS, Nginx

کانتینرسازی: Docker و Docker Compose

وابستگی‌ها: فهرست شده در requirements.txt (fastapi, Uvicorn, sqlalchemy, pydantic)

➤ پیش نیازها

Docker و Docker Compose

Python 3.13

Pip (برای نصب بسته‌های پایتون)

Git

➤ نصب و راه اندازی:

فایل **main.py** هسته‌ی اصلی پروژه است و شامل پیاده‌سازی کامل **API** با استفاده از **FastAPI** می‌باشد. این فایل وظایف زیر را بر عهده دارد:

تعریف مدل‌های داده‌ای برای سه موجودیت اصلی پروژه **Student**، **Teacher** و **Course** با استفاده از **SQLModel**

اعتبارسنجی کامل مقادیر ورودی با استفاده از **@validator** برای اطمینان از صحت و سازگاری داده‌ها با نیازمندی‌های پروژه (مثلاً فرمت کد ملی شماره دانشجویی، تاریخ تولد شمسی و ...).

پیاده‌سازی عملیات **CRUD** (ایجاد، خواندن، ویرایش و حذف) برای هر سه موجودیت از طریق **endpoint** های **FastAPI**.

تعریف **session** و اتصال به پایگاه داده **SQLite** برای مدیریت داده‌ها.

ساختاردهی **API** با استفاده از **@app.get**، **@app.post**، **@app.put**، **@app.delete** برای مسیرهای **RESTful**.

این فایل با اجرای مستقیم سرور **FastAPI**، سرویس‌دهی **API** را فعال می‌کند و آماده است تا روی سرور لینوکسی با **Docker** و **Nginx** مستقر شود.

این فایل بدون نیاز به فایل‌های اضافی قابل اجراست و پایگاه داده را در اولین اجرا به‌صورت خودکار ایجاد می‌کند. (تمام خطاهای احتمالی ناشی از ورود داده‌های نامعتبر از طریق **FastAPI** با پیام مناسب به کاربر برگشت داده می‌شود).

✧ در واقع فعالیت های 1 تا 4 پروژه ما را در بر میگیرد

برای اجرای بک اند دستور زیر اجرا می‌شود:

```
uvicorn main:app --host 0.0.0.0 --port 8000
```

5. داکرایز کردن پروژه و انتقال به سرور

در این مرحله، پروژه **FastAPI** به کمک **Docker** و **Docker Compose** داکرایز و روی سرور لینوکسی اجرا شد. مراحل به شرح زیر هستند

1 به‌روزرسانی سرور

با استفاده از ابزار **PutTY** به سرور متصل و دستورات زیر را اجرا کردیم تا سیستم به‌روز شود:

`sudo apt update`

`sudo apt upgrade`

```
root@id-345247: ~  
Get:21 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [157  
kB]  
Get:22 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [2  
1.6 kB]  
Get:23 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Package  
s [1,131 kB]  
Get:24 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-e  
n [235 kB]  
Get:25 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Compone  
nts [212 B]  
Get:26 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages  
[842 kB]  
Get:27 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en  
[104 kB]  
Get:28 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Component  
s [52.2 kB]  
Get:29 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Compone  
nts [212 B]  
Fetched 8,552 kB in 4s (2,052 kB/s)  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
2 packages can be upgraded. Run 'apt list --upgradable' to see them.  
root@id-345247:~# apt upgrade
```

2. نصب Docker و Docker Compose

برای نصب داکر و داکر کامپوز از دستورات زیر استفاده کرد

`sudo curl -fsSL https://get.docker.com | sh`

3. تست موفق نصب Docker

برای اطمینان از نصب موفق داکر، از دستور زیر استفاده کردم:

`sudo docker run hello-world`

```
root@id-345247: ~  
No user sessions are running outdated binaries.  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@id-345247:~# curl -fsSL https://get.docker.com | sudo sh  
Executing docker install script, commit: 53a22f61c0628e58e1d6680b49e82993d304b  
49  
Warning: the "docker" command appears to already exist on this system.  
If you already have Docker installed, this script can cause trouble, which is  
why we're displaying this warning and provide the opportunity to cancel the  
installation.  
If you installed the current Docker package using this script and are using it  
again to update Docker, you can ignore this message, but be aware that the  
script resets any custom changes in the deb and rpm repo configuration  
files to match the parameters passed to the script.  
You may press Ctrl+C now to abort this script.  
sleep 20  
C  
root@id-345247:~# ^C  
root@id-345247:~# sudo docker run hello-world
```

نتیجه اجرای دستور پیام موفقیت آمیز بود.

4. آماده سازی فایل های Docker

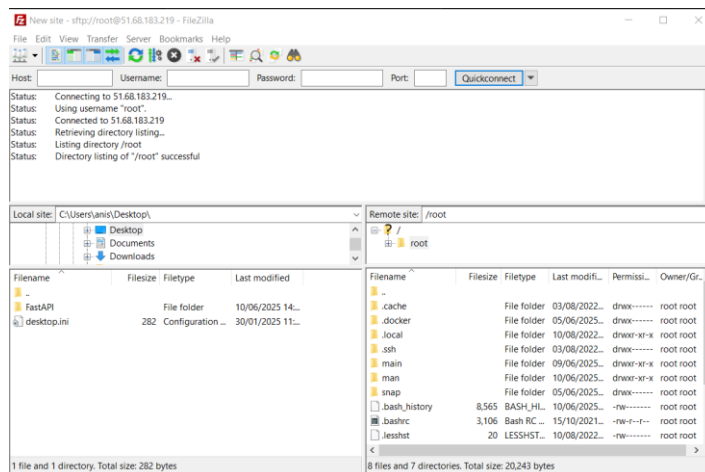
در محیط توسعه، دو فایل اصلی زیر را ایجاد کردم:

Dockerfile برای ساخت image پروژه FastAPI

docker-compose.yml برای اجرای پروژه و پایگاه داده به صورت همزمان

5. انتقال فایل ها به سرور

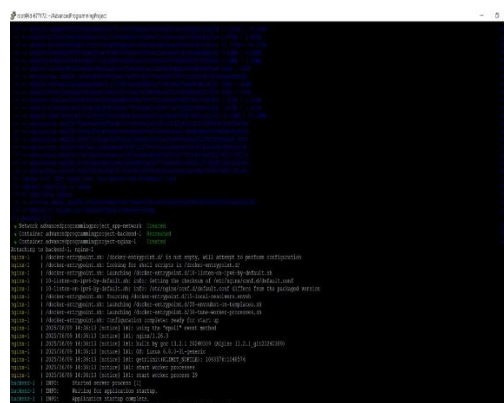
با استفاده از نرم افزار **FileZilla**، فایل های پروژه و فایل های داکر به سرور منتقل شدند.



6. اجرای کانترینرها روی سرور

در نهایت در مسیر پروژه روی سرور، کانترینرها را با دستور زیر اجرا کردم:

```
sudo docker-compose up --build -d
```



بعد از اتمام این کار ها داکرایز کردن انجام شد و در **swagger** به آدرس زیر به درستی بالا میاید.

<http://anisdelfani.ir:8080/docs>

7. خرید یک سرور مجازی لینوکسی

یک سرور مجازی هاست سایت پارس وی دی ای خریداری شد و تمام ابزار های مورد نیاز از طریق پووتی بر روی آن نصب شد (به عنوان مثال داکر و ...) که در فعالیت فوق دیدیم.



8. استفاده از Nginx

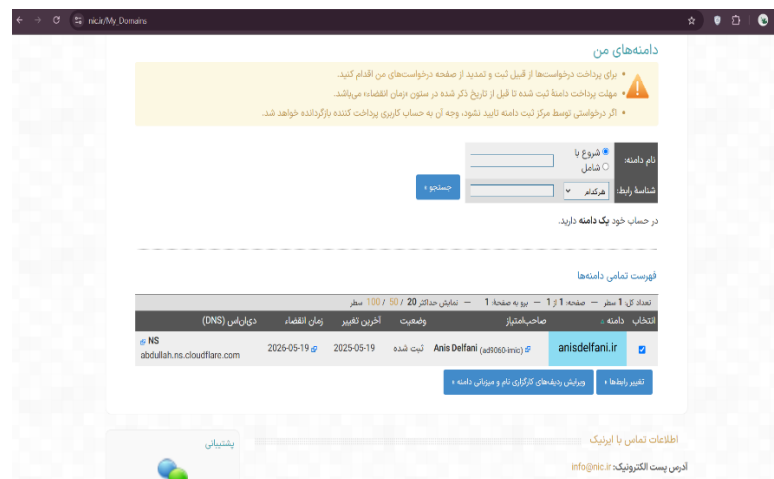
و اجرای nginx باید کار های زیر را انجام دهیم و این تغییرات برای این است که بدون وارد کردن پورت به لینک خود در سووگر وصل بشیم یعنی به صورت :

<http://anisdelfani.ir/docs>

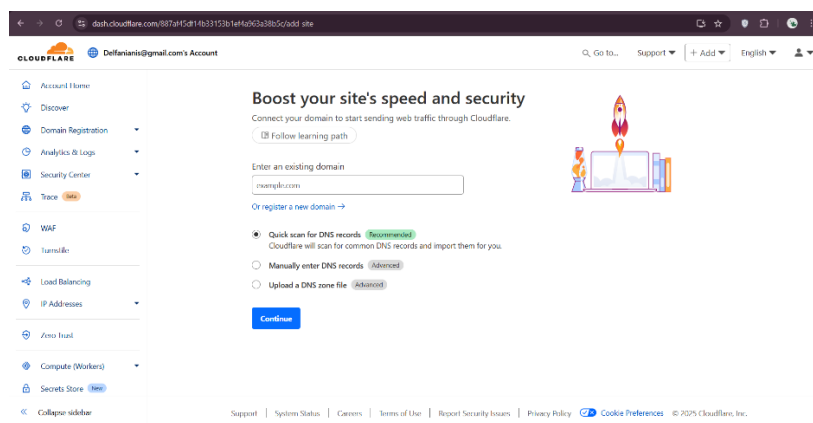
ابتدا با ایجاد کردن دو فایل با نام های Dockerfile.nginx و nginx.conf این فایل هارا با فایل زیلا بر روی سرور مجازی (هاست) اپلود میکنیم و اجرای دوباره docker-compose حال لینک بدون پورت بالا می آید.

9. خرید دامین از سات ایرنیک و اتصال به couldflare

یک دامنه از سایت ایرنیک تهیه کردیم



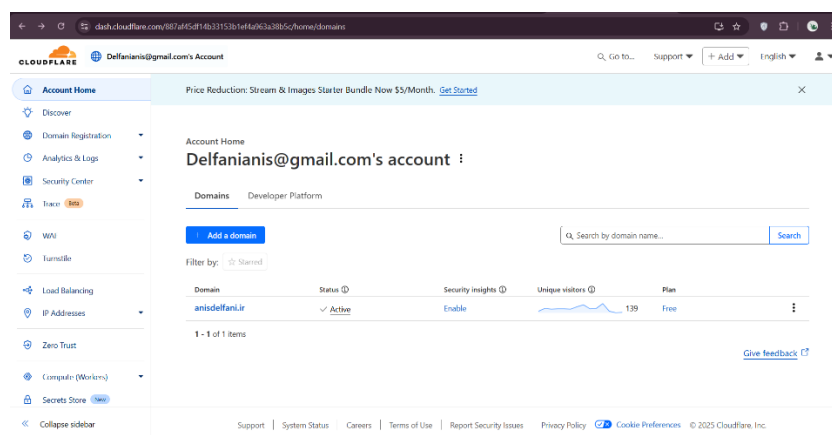
پس از خرید دامین خود باید آن را به cloudflare اتصال دهیم برای این کار وارد کلود فلر شده در این قسمت نام دامین خود را وارد میکنیم



و پس از فرستادن دامین خود رمز هایی که کلود فلر برای ورود میدهد را در ایرنیک وارد میکنیم

پس وارد کردن نام و رمز ها اتصال را میزنیم و اتصال بر قرار میشود

پس دو روز سرور درخواست را پذیرفته و آن را اکتیو میکند



بعد از انجام فرانت اند:

مراحل را مانند قبل پیش رفتیم باین تفاوت که فایل های فرانت (html,css,js,..) به فایل زیلا اضافه شد.

اکنون دستور زیر را در putty پیاده می کنیم:

```
cd main
```

سپس دستورات زیر را به ترتیب وارد می کنیم:

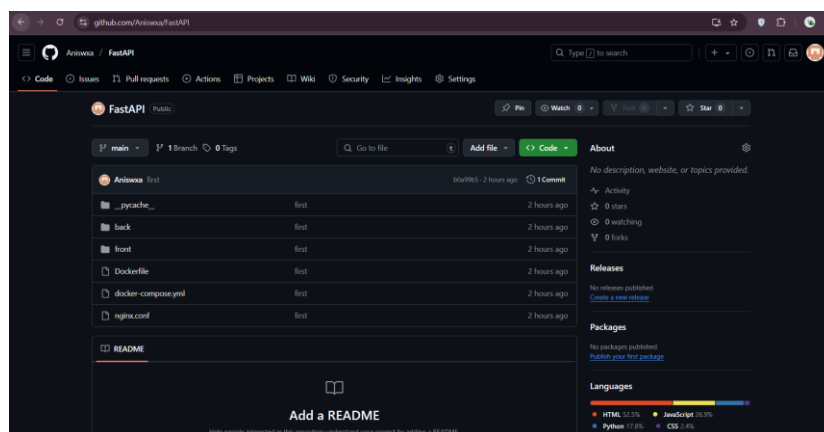
```
docker-compose down
```

```
docker-compose up --build
```

پس از تمام این کار ها در فعالیت ششم باید تمام فایل ها را در بستر GitHub

از طریق IDE و push کنیم:

<https://github.com/Aniswxa/FastAPI>



12. در اخر همه API ها بر روی دامنه قابل دسترس است و ارتباط فرانت اند و بک اند درست کار می کند و با آدرس زیر سایت بالا

می آید:

<http://anisdelfani.ir>