



به نام خدا

تمرین اول درس پایگاه داده پیشرفته

دانشجو:

پریسا مبارک

شماره دانشجویی:

۴۰۲۱۱۴۱۵۰۰۶

استاد مربوطه:

دکتر عبدالرضا رشنو

آذر ۱۴۰۲

شرح کار:

سه جدول student و teacher و lesson داریم. می‌خواهیم با استفاده از پایگاه داده sqlie و fastapi مدل‌ها را پیاده‌سازی کنیم

در فایل models.py:

مطابق با فایل‌های ضمیمه شده مدل‌ها را پیاده‌سازی کردم. در این فایل ۴ مدل student و teacher و lesson داریم که هر کدام فیلدهای مخصوص خود را دارند.

```
API test_main.http  main.py  database.py  schemas.py  models.py ×
1  from database import Base
2  from sqlalchemy import Column, Integer, String
3
4  9 usages
5  class Student(Base):
6      __tablename__ = 'student'
7
8      stid = Column(Integer, primary_key=True, index=True)
9      fname = Column(String)
10     lname = Column(String)
11     father = Column(String)
12     birth = Column(String)
13     ids = Column(String, unique=True)
14     borncity = Column(String)
15     addres = Column(String)
16     postalcode = Column(Integer)
17     cphone = Column(Integer)
18     hphone = Column(Integer)
19     department = Column(String)
20     major = Column(String)
21     married = Column(String)
22     id = Column(Integer, unique=True)
23     scourseids = Column(Integer)
24     lids = Column(Integer)
25
26     9 usages
27 class Teacher(Base):
28     __tablename__ = 'teacher'
29
30     lid = Column(Integer, primary_key=True, index=True)
31     fname = Column(String)
```

شکل ۱. یک نمونه از پیاده سازی مدل student در فایل models.py

در فایل schemas.py:

کلاس های مربوط به هر مدل را نیز پیاده سازی کردم.

```

2 usages
class StudentBase(BaseModel):
    fname: str
    lname: str
    father: str
    birth: str
    ids: str
    borncity: str
    addres: str
    postalcode: int
    cphone: int
    hphone: int
    department: str
    major: str
    married: str
    scourseids: int
    lids: int

```

شکل ۲. یک کلاس برای جدول دانشجو

در ادامه برای استفاده از responsive model ها یک سری فیلد ها را در کلاس های جداگانه نیز گذاشتم (مانند کلاس ReadStu) که در ادامه کاربرد آن ها را بررسی میکنیم.

```

1 usage
class StudentRead(BaseModel):
    stid: int
    fname: str
    lname: str
    father: str

```

شکل ۳. ایجاد یک کلاس برای استفاده از responsv model ها در جدول دانشجو

همچنین در این فایل اعتبارسنجی های مربوط به فیلدهای هر جدول را انجام دادم.
به عنوان مثال برای فیلد birth در جدول student میخواهم مطابق با الگوی منظم تاریخ شمسی باشد:

```
@validator('birth')
def validate_birth(cls, birth):
    date_pattern = "^1[3-4][0-9]{2}/((0[0-9])|(1[0-2]))/(((0[0-2])|(3[0-1]))$)"
    match = re.match(date_pattern, birth)
    if not match:
        raise ValueError("تاریخ تولد نامعتبر است.")
    return birth
```

شکل ۴. یک نمونه اعتبارسنجی در جدول دانشجو

در فایل database.py:

در این فایل به پایگاه داده sqlite متصل میشویم.

```
API test_main.http  main.py  database.py  schemas.py  models.py
1  from sqlalchemy import create_engine
2  from sqlalchemy.ext.declarative import declarative_base
3  from sqlalchemy.orm import sessionmaker
4  SQLALCHEMY_DATABASE_URL = 'sqlite:///./test.db'
5  engine = create_engine(SQLALCHEMY_DATABASE_URL, connect_args={'check_same_thread': False})
6  SessionLocal = sessionmaker(bind=engine)
7  Base = declarative_base()
8
```

شکل ۵. محتوای فایل database.py

در فایل main.py:

در این فایل برای عملیات درج و حذف و آپدیت و خواندن در هر جدول یک api طراحی میکنیم.
به عنوان مثال برای جدول student:

```
t_main.http  main.py × database.py schemas.py models.py

from fastapi import FastAPI, Depends, HTTPException
from sqlalchemy.orm import Session
from database import engine, SessionLocal
import schemas, models

models.Base.metadata.create_all(bind=engine)
app = FastAPI()

12 usages
def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()

@app.post(path: '/RegStu/', response_model=schemas.Student)
def create_student(student: schemas.StudentCreate, db: Session=Depends(get_db)):
    db_student=db.query(models.Student).filter(models.Student.ids ==student.ids).first()
    if db_student:
        raise HTTPException(status_code=400, detail='already exists')
    student=models.Student(fname=student.fname, lname=student.lname, father=student.father, birth:
    db.add(student)
    db.commit()
    db.refresh(student)
    return student
```

شکل ۶. یک api برای عمل درج رکورد در جدول دانشجو

```

26
27 @app.get(path: '/RegStu/{student_stid}', response_model=schemas.StudentRead)
28 def read_student(student_stid: int, db: Session = Depends(get_db)):
29     db_student = db.query(models.Student).filter(models.Student.stid == student_stid).first()
30     if db_student is None:
31         raise HTTPException(status_code=404, detail='STUDENT NOT FOUND')
32     return db_student
33
34
35
36 @app.delete('/DelStu/{student_stid}')
37 def delete_student(student_stid: int, db: Session = Depends(get_db)):
38     db_student = db.query(models.Student).filter(models.Student.stid == student_stid).first()
39     if db_student is None:
40         raise HTTPException(status_code=404, detail='STUDENT NOT FOUND')
41     db.delete(db_student)
42     db.commit()
43     return {"message": f"Student with stid {student_stid} has been deleted."}
44

```

شکل ۷. یک api برای عمل خواندن رکورد و یک api برای عمل حذف رکورد از جدول دانشجو

```

@app.post(path: '/UpStu/{student_std}', response_model=schemas.Student)
def update_student(student_std: int, student: schemas.StudentCreate, db: Session = Depends(get_db)):
    db_student = db.query(models.Student).filter(models.Student.std == student_std).first()
    if db_student is None:
        raise HTTPException(status_code=404, detail="Student not found")

    # تنظیم مستقیم فیلدها از اطلاعات جدید دانشجو
    db_student.fname = student.fname
    db_student.lname = student.lname
    db_student.father = student.father
    db_student.birth = student.birth
    db_student.ids = student.ids
    db_student.borncity = student.borncity
    db_student.addres = student.addres
    db_student.postalcode = student.postalcode
    db_student.cphone = student.cphone
    db_student.hphone = student.hphone
    db_student.department = student.department
    db_student.major = student.major
    db_student.married = student.married
    db_student.scourseids = student.scourseids
    db_student.lids = student.lids
    db_student.id = student.id

    # ذخیره تغییرات در پایگاه داده
    db.commit()

    # بازخوانی دانشجو برای به‌روزرسانی اطلاعات
    db.refresh(db_student)

```

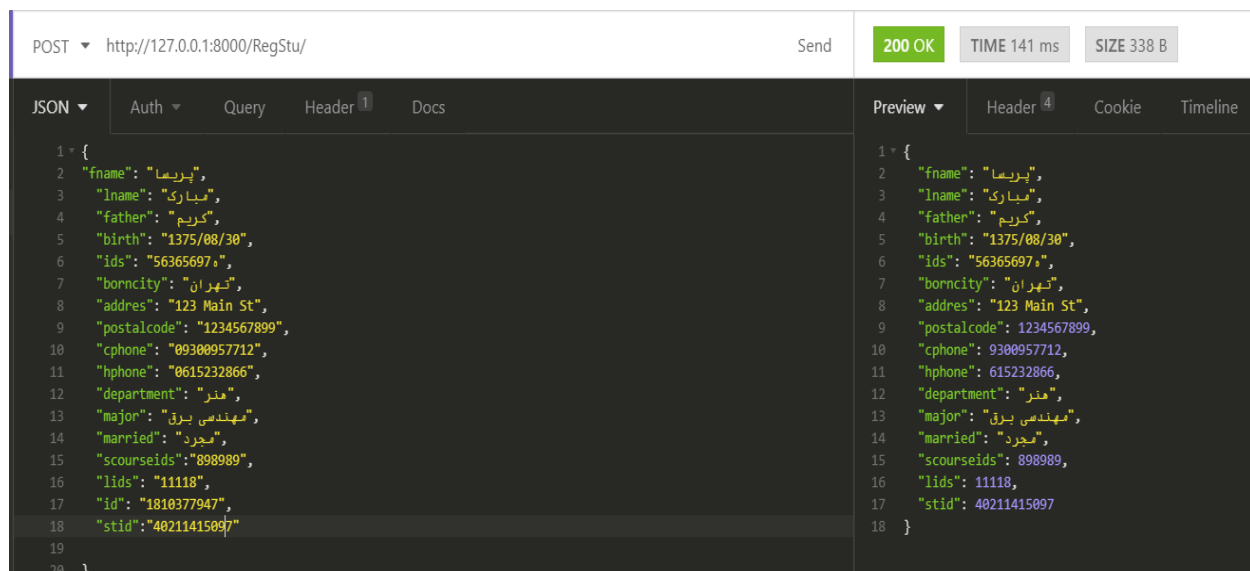
شکل ۸. یک api برای عمل آپدیت رکورد در جدول دانشجو

در فایل های ضمیمه شده تمام کدها به طور کامل برای هر جدول وجود دارد. که به دلیل طولانی بودن تنها خلاصه ای از آن ها را در بالا نشان دادم.

تست api ها:

در اینجا برای تست api ها از ابزار insomnia استفاده میکنیم:

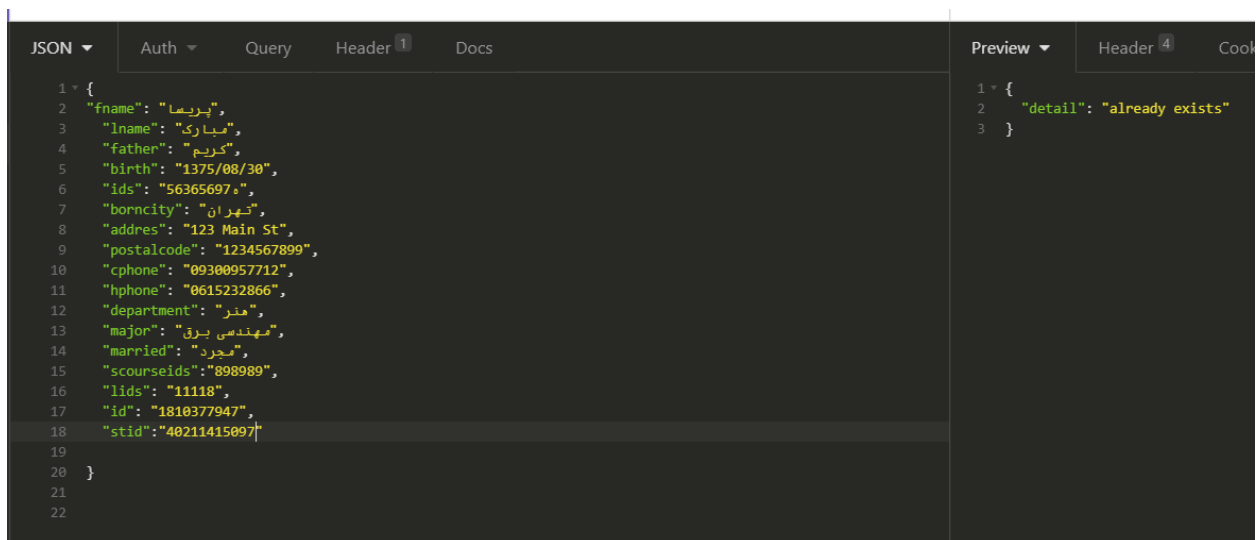
درج رکورد در جدول دانشجو:



شکل ۹. درج رکورد در جدول دانشجو

همانطور که مشاهده میکنیم با متد post و از طریق url نوشته شده توانستم یک رکورد به جدول دانشجو اضافه کنم.

حال اگر بخواهیم دوباره این رکورد را وارد کنیم به ما میگوید رکورد وجود دارد:



شکل ۱۰. درج رکورد تکراری در جدول دانشجو

خواندن رکورد از جدول دانشجو:

The screenshot shows a REST client interface. The top bar indicates a GET request to `http://127.0.0.1:8000/RegStu/40211415097` with a status of **200 OK** and a response time of **16 ms**. The left pane shows the JSON response, and the right pane shows a preview of the same JSON.

```
1 {
2   "fname": "پریم",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377947",
18  "stid": "40211415097"
19 }
20
21
```

شکل ۱۱. خواندن رکورد از جدول دانشجو

مطابق تصویر بالا با استفاده از متد `get` از طریق `url` بالا و از طریق شماره `stid` که در آخر `url` نوشتیم، توانستیم یک رکورد را بخوانیم و ۴ فیلد اول جدول دانشجو را به عنوان خروجی گرفتیم. اگر بخواهیم رکوردی که وجود ندارد را بخوانیم به ما میگوید دانشجو پیدا نمیشود:

The screenshot shows a REST client interface. The top bar indicates a GET request to `http://127.0.0.1:8000/RegStu/4344` with a status of **404 Not Found** and a response time of **0 ms**. The left pane shows the JSON response, and the right pane shows a preview of the same JSON.

```
1 {
2   "fname": "پریم",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377947",
18  "stid": "40211415097"
19 }
20
21
```

```
1 {
2   "detail": "STUDENT NOT FOUND"
3 }
```

شکل ۱۲. خواندن رکوردی که در جدول دانشجو وجود ندارد

آپدیت رکورد در جدول دانشجو:

The screenshot shows a REST client interface with a POST request to `http://127.0.0.1:8000/UpStu/40211415097`. The request body is a JSON object with the following fields: `fname`, `lname`, `father`, `birth`, `ids`, `borncity`, `addres`, `postalcode`, `cphone`, `hphone`, `department`, `major`, `married`, `scourseids`, `lids`, `id`, and `stid`. The response status is `200 OK` with a time of `281 ms`. The response body is a JSON object with the same fields, but the `stid` field is updated to `40211415097`.

```
1 {
2   "fname": "پریمیا",
3   "lname": "امیری",
4   "father": "امیر",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377947",
18  "stid": "40211415097"
19 }
20 }
```

```
1 {
2   "fname": "پریمیا",
3   "lname": "امیری",
4   "father": "امیر",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "هنر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415097"
18 }
```

شکل ۱۳. آپدیت یک رکورد در جدول دانشجو

طبق تصویر بالا فیلدهای `lname` و `father` رکورد مدنظر را آپدیت کردم.

حذف رکورد از جدول دانشجو:

The screenshot shows a REST client interface with a DELETE request to `http://127.0.0.1:8000/DelStu/40211415097`. The request body is a JSON object with the same fields as the previous request. The response status is `200 OK` with a time of `109 ms` and a size of `61 B`. The response body is a JSON object with a `message` field: `"Student with stid 40211415097 has been deleted."`.

```
1 {
2   "fname": "پریمیا",
3   "lname": "امیری",
4   "father": "امیر",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377947",
18  "stid": "40211415097"
19 }
20 }
```

```
1 {
2   "message": "Student with stid 40211415097 has been
3   deleted."
4 }
```

شکل ۱۴. حذف یک رکورد از جدول دانشجو

طبق تصویر بالا رکورد مدنظر را از طریق متد delete واز طریق url بالا حذف کردم.

حال اگر بخوایم رکورد را بخوانیم چون رکورد حذف شده است خطا میدهد.مطابق تصویر زیر:

GET http://127.0.0.1:8000/RegStu/40211415097

Send 404 Not Found TIME 0 ms SIZE 30 B

JSON Auth Query Header 1 Docs

```
1 {
2   "fname": "پریمیا",
3   "lname": "امیری",
4   "father": "امیر",
5   "birth": "1375/08/30",
6   "ids": "56365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377947",
18  "stid": "40211415097"
19 }
20 }
```

Preview Header 4 Cookie Timeline

```
1 {
2   "detail": "STUDENT NOT FOUND"
3 }
```

شکل ۱۵. خواندن رکوردی که حذف شده است از جدول دانشجو

همانطور که مشاهده میکنیم در خروجی به ما میگوید که دانشجو وجود ندارد.

اعتبارسنجی فیلدها در جدول دانشجو:

همانطور که گفتیم در فایل schemas.py اعتبارسنجی شده اند که نمونه ای از آن ها را در ادامه مشاهده

میکنید. به عنوان مثال برای فیلدهای fname,lname,father

POST http://127.0.0.1:8000/RegStu/

Send 422 Unprocessable Entity TIME 16 ms SIZE 259

JSON Auth Query Header 1 Docs

```
1 {
2   "fname": "sima",
3   "lname": "سماری",
4   "father": "کرم",
5   "birth": "1375/08/30",
6   "ids": "56065697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377940",
18  "stid": "40211415000"
19 }
20 }
```

Preview Header 4 Cookie Timeline

```
1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "fname"
8       ],
9       "msg": "Value error, نام باید فقط شامل حروف فارسی باشد",
10      "input": "sima",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }
```

شکل ۱۶. درج رکورد با مقادیر نامعتبر برای فیلدهای lname, fname, father

طبق تصویر بالا، نام را انگلیسی نوشتم و خطا داد. وقتی خطا را رفع کردم و نام را فارسی نوشتم رکورد بدون خطا اضافه شد:

POST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 125 ms SIZE: 100 B

JSON Auth Query Header 1 Docs

```
1 {
2   "fname": "سید",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "56065697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "هنر",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377940",
18  "stid": "40211415000"
19 }
20 }
21
22
```

Preview Header 4 Cook

```
1 {
2   "fname": "سید",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "56065697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "هنر",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415000"
18 }
```

شکل ۱۷. رفع خطای مربوط به نامعتبر بودن فیلدهای lname, fname, father

برای فیلدهای دیگر مثل تاریخ تولد و محل تولد و آدرس و شناسه کدملی و آدرس و کدپستی و نام دانشکده ورشته و وضعیت تاهل و کد درس و کد اساتید و کد ملی و شماره دانشجویی و سایر فیلدها نیز اعتبارسنجی کردم. در ادامه برای بررسی اعتبارسنجی هر کدام از فیلدها یک تصویر از اطلاعات نادرست و خطای آن و همچنین یک تصویر از اطلاعات درست و درج رکورد بدون خطا مشاهده میکنید:

POST http://127.0.0.1:8000/RegStu/ Send

422 Unprocessable Entity TIME 16 ms SIZE 215

JSON Auth Query Header 1 Docs

```
1 {
2   "fname": "فربیا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "16365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377911",
18  "stid": "40211415011"
19 }
20
21
22
```

Preview Header 4 Cookie Timeline

```
1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "birth"
8       ],
9       "msg": "Value error, نامعتبر است تاریخ تولد نامعتبر.",
10      "input": "135/08/30",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }
```

شکل ۱۸. درج رکورد با مقادیر نامعتبر برای فیلد birth

POST http://127.0.0.1:8000/RegStu/ Send

200 OK TIME 125 ms SIZE 338

JSON Auth Query Header 1 Docs

```
1 {
2   "fname": "فربیا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "16365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377911",
18  "stid": "40211415011"
19 }
20
21
22
```

Preview Header 4 Cookie

```
1 {
2   "fname": "فربیا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "16365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415011"
18 }
```

شکل ۱۹. رفع خطای مربوط به نامعتبر بودن فیلد birth

POST
http://127.0.0.1:8000/RegStu/
Send
422 Unprocessable Entity
TIME 0 ms
SIZE 320 B

JSON
Auth
Query
Header
Docs

```

1 {
2   "fname": "کریم",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "16365697",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377912",
18  "stid": "40211415012"
19 }
20 }
21
22

```

Preview
Header
Cookie
Timeline

```

1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "ids"
8       ],
9       "msg": "Value error, باید شماره شناسایی نامعتبر است. یک حرف فارسی و یک عدد 2 رقمی باشد",
10      "input": "16365697",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }

```

شکل ۲۰. درج رکورد با مقادیر نامعتبر برای فیلد ids

POST
http://127.0.0.1:8000/RegStu/
Send
200 OK
TIME 172 ms
SIZE 336 B

JSON
Auth
Query
Header
Docs

```

1 {
2   "fname": "کریم",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "67676767",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377912",
18  "stid": "40211415012"
19 }
20 }
21
22

```

Preview
Header
Cookie
Timeline

```

1 {
2   "fname": "کریم",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "67676767",
7   "borncity": "تهران",
8   "addres": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "منر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415012"
18 }

```

شکل ۲۱. رفع خطای مربوط به نامعتبر بودن فیلد ids

POST http://127.0.0.1:8000/RegStu/ Send 422 Unprocessable Entity TIME 0 ms SIZE 287 B

JSON Auth Query Header Docs

```
1 {
2   "fname": "سارا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "67675767",
7   "borncity": "تهران",
8   "address": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377913",
18  "stid": "40211415013"
19 }
20 }
21
22
```

Preview Header 4 Cookie Timeline

```
1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "borncity"
8       ],
9       "msg": "Value error, باید نامعتبر است. شهر محل تولد نامعتبر است. یکی از مراکز استان های کشور باشد",
10      "input": "ته",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }
```

شکل ۲۲. درج رکورد با مقادیر نامعتبر برای فیلد borncity

POST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 203 ms SIZE 100 B

JSON Auth Query Header Docs

```
1 {
2   "fname": "سارا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "67675767",
7   "borncity": "تهران",
8   "address": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377913",
18  "stid": "40211415013"
19 }
20 }
21
22
```

Preview Header 4 Cookie

```
1 {
2   "fname": "سارا",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "67675767",
7   "borncity": "تهران",
8   "address": "123 Main St",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377913",
18  "stid": "40211415013"
19 }
20 }
```

شکل ۲۳. رفع خطای مربوط به نامعتبر بودن فیلد borncity

POST http://127.0.0.1:8000/RegStu/ Send 422 Unprocessable Entity TIME 0 ms SIZE 262

JSON Auth Query Header 1 Docs Preview Header 4 Cookie Timeline

```
1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52570067",
7   "borncity": "تهران",
8   "addres": "123 Main Sttttt",
9   "postalcode": "12345",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منتر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377916",
18  "stid": "40211415016"
19 }
20
21
22
```

```
1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "postalcode"
8       ],
9       "msg": "Value error, باید شامل یک عدد 10 رقمی باشد",
10      "input": "12345",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }
```

شکل ۲۶. درج رکورد با مقادیر نامعتبر برای فیلد postalcode

OST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 141 ms SIZE 339

JSON Auth Query Header 1 Docs Preview Header 4 Cookie

```
1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52570067",
7   "borncity": "تهران",
8   "addres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منتر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377916",
18  "stid": "40211415016"
19 }
20
21
22
```

```
1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52570067",
7   "borncity": "تهران",
8   "addres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "منتر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415016"
18 }
```

شکل ۲۷. رفع خطای مربوط به نامعتبر بودن فیلد postalcode

POST http://127.0.0.1:8000/RegStu/ Send 422 Unprocessable Entity TIME 16 ms SIZE 277

JSON Auth Query Header Docs Preview Header Cookie Timeline

```

1 {
2   "fname": "علی",
3   "lname": "میبارکی",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52500067",
7   "borncity": "تهران",
8   "address": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "ه",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377917",
18  "stid": "40211415017"
19 }
20 }
21
22

```

```

1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "department"
8       ],
9       "msg": "Value error, باید نام دانشگاه نامعتبر است.",
10      "input": "ه",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    }
16  ]
17 }

```

شکل ۲۸. درج رکورد با مقادیر نامعتبر برای فیلد department

POST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 125 ms SIZE 339 B

JSON Auth Query Header Docs Preview Header Cookie

```

1 {
2   "fname": "علی",
3   "lname": "میبارکی",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52500067",
7   "borncity": "تهران",
8   "address": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "مهر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377917",
18  "stid": "40211415017"
19 }
20 }
21
22

```

```

1 {
2   "fname": "علی",
3   "lname": "میبارکی",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "52500067",
7   "borncity": "تهران",
8   "address": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "مهر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415017"
18 }

```

شکل ۲۹. رفع خطای مربوط به نامعتبر بودن فیلد department

POST http://127.0.0.1:8000/RegStu/ Send 422 Unprocessable Entity TIME 16 ms SIZE 417

JSON Auth Query Header 1 Docs Preview Header 4 Cookie Timeline

```
1 {
2   "fname": "علی",
3   "lname": "میاری",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "50500067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منر",
13  "major": "مهندسی",
14  "married": "م",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377918",
18  "stid": "40211415018"
19 }
20
21
22
```

```
1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "major"
8       ],
9       "msg": "Value error, نامعتبر است",
10      "input": "مهندسی",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    },
16    {
17      "type": "value_error",
18      "loc": [
19        "body",
20        "married"
21      ],
22      "msg": "Value error, نامعتبر است",
23      "input": "م",
24      "ctx": {
25        "error": {}
26      },
27      "url": "https://errors.pydantic.dev/2.5/v/value_error"
28    }
29  ]
30 }
```

شکل ۳۰. درج رکورد با مقادیر نامعتبر برای فیلدهای major , married

POST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 110 ms SIZE 339

JSON Auth Query Header 1 Docs Preview Header 4 Cookie

```
1 {
2   "fname": "علی",
3   "lname": "میاری",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "50500067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "id": "1810377918",
18  "stid": "40211415018"
19 }
20
21
22
```

```
1 {
2   "fname": "علی",
3   "lname": "میاری",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "50500067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "منر",
13  "major": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "898989",
16  "lids": "11118",
17  "stid": "40211415018"
18 }
```

شکل ۳۱. رفع خطای مربوط به نامعتبر بودن فیلدهای major , married

POST http://127.0.0.1:8000/RegStu/ Send 422 Unprocessable Entity TIME 0 ms SIZE 477

JSON Auth Query Header Docs Preview Header Cookie Timeline

```

1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "50900067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "0615232866",
12  "department": "منزل",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "8989",
16  "lids": "11118",
17  "id": "181037791",
18  "stid": "4021141501"
19 }
20
21
22

```

```

1 {
2   "detail": [
3     {
4       "type": "value_error",
5       "loc": [
6         "body",
7         "id"
8       ],
9       "msg": "شماره شناسنامه نامعتبر است. باید شامل یک عدد 10 رقمی باشد",
10      "input": "181037791",
11      "ctx": {
12        "error": {}
13      },
14      "url": "https://errors.pydantic.dev/2.5/v/value_error"
15    },
16    {
17      "type": "value_error",
18      "loc": [
19        "body",
20        "stid"
21      ],
22      "msg": "کد دانشجویی نامعتبر است",
23      "input": "4021141501",
24      "ctx": {
25        "error": {}
26      },
27      "url": "https://errors.pydantic.dev/2.5/v/value_error"
28    }
29  ]
30 }

```

شکل ۳۲. درج رکورد با مقادیر نامعتبر برای فیلدهای id,stid

POST http://127.0.0.1:8000/RegStu/ Send 200 OK TIME 187 ms SIZE 31

JSON Auth Query Header Docs Preview Header Cookie

```

1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "60600067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "09300957712",
11  "hphone": "615232866",
12  "department": "منزل",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "8989",
16  "lids": "11118",
17  "id": "1801037766",
18  "stid": "40211415059"
19 }
20
21
22

```

```

1 {
2   "fname": "علی",
3   "lname": "مبارک",
4   "father": "کریم",
5   "birth": "1375/08/30",
6   "ids": "60600067",
7   "borncity": "تهران",
8   "adres": "123 Main Sttttt",
9   "postalcode": "1234567899",
10  "cphone": "9300957712",
11  "hphone": "615232866",
12  "department": "منزل",
13  "majon": "مهندسی برق",
14  "married": "مجرد",
15  "scourseids": "8989",
16  "lids": "11118",
17  "stid": "40211415059"
18 }

```

شکل ۳۳. رفع خطای مربوط به نامعتبر بودن فیلدهای id,stid

در ادامه برای فیلدهای جدوال teacher و lesson نیز به همین صورت اعتبارسنجی میکنیم. که در فایل های ضمیمه شده کدهای مربوط به آن ها نیز وجود دارد و میتوان تست کرد.

پایان