Important: Beta Release coming soon!

Source code for api.models.user

University of Nebraska Tutoring Portal

CSLC PORTAL BACKEND

About

API Endpoints

Database Models

API Scripts

API Config

CSLC PORTAL FRONTEND

About

```python
from django.db import models
from django.db.models.query import QuerySet

from .course import Course


class StudentManager(models.Manager):

    def get_students(self) -> QuerySet:
        return super().get_queryset().filter(is_tutor=False).filter(is_admin=False)

    def get_student(self, name: str) -> QuerySet:
        return (
            super()
            .get_queryset()
            .filter(is_tutor=False)
            .filter(is_admin=False)
            .filter(name)
        )


class TutorManager(models.Manager):

    def get_tutors(self) -> QuerySet:
        return super().get_queryset().filter(is_tutor=True)

    def get_tutor(self, name: str) -> QuerySet:
        return super().get_queryset().filter(is_tutor=True).filter(name)
```

```python
class AdminManager(models.Manager):

    def get_admins(self) -> QuerySet:
        return super().get_queryset().filter(is_admin=True)




class User(models.Model):
    """
    Generic user model for all active users of the application. Flags are used
    to indicate whether a user is an admin, a tutor, or a regular student. This
    currently will hold all data, such as tutor and admin specific data, but
    this might lead to empty fields within many of the objects in the table.
    This will be updated in the future to allow specific roles to be relations
    in the database, thus preventing empty fields (i.e. empty cells).
    """

    student_nuid = models.BigIntegerField(
        primary_key=True, unique=True, blank=False, null=False, default=1
    )
    courses_tutoring = models.ManyToManyField(
        Course, related_name="usertocoursetutored", blank=True
    )
    courses_taken = models.ManyToManyField(
        Course, related_name="usertocoursetaken", blank=True
    )
    name = models.CharField(max_length=100)
    is_active = models.BooleanField(default=False)
    is_working = models.BooleanField(default=False)
    is_tutor = models.BooleanField(default=False)
    is_admin = models.BooleanField(default=False)
    user_bio = models.TextField(blank=True, max_length=500)
    email = models.EmailField(blank=False, unique=True)
    MSOID = models.CharField(max_length=75, unique=True)

    generic = models.Manager()
    student = StudentManager()
    tutor = TutorManager()
    admin = AdminManager()
```

```python
    def __str__(self) -> str:
        return str(self.name)
```