```
# import flask_sqlalchemy and create db instance
from flask_sqlalchemy import SQLAlchemy
db = SQLAlchemy()
# hashing
from flask_bcrypt import Bcrypt
bcrypt = Bcrypt()
from datetime import datetime


def connect_db(app):
    """Connect to database."""
    db.app = app
    db.init_app(app)


class User(db.Model):
    __tablename__ = "users"
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    username = db.Column(db.String(20), nullable=False,unique=True)
    password = db.Column(db.Text, nullable=False)
    email = db.Column(db.String(50), nullable=False, unique=True)
    first_name = db.Column(db.String(30), nullable=False)
    last_name = db.Column(db.String(30), nullable=False)
    saved_stories = db.relationship(
        'Story', cascade="all, delete-orphan")
    queries = db.relationship(
        'Query', cascade="all, delete-orphan")

    @classmethod
```

```
    def register(cls, username, pwd, email, first_name, last_name):
        hashed = bcrypt.generate_password_hash(pwd)
        hashed_utf8 = hashed.decode("utf8")
        new_user = cls(username=username, password=hashed_utf8,
                       email=email, first_name=first_name, last_name=last_name)
        db.session.add(new_user)
        return new_user

    @classmethod
    def authenticate(cls, username, pwd):
        u = User.query.filter_by(username=username).first()
        if u and bcrypt.check_password_hash(u.password, pwd):
            return u
        else:
            return False

    def __repr__(self):
        return f"<ID: {self.id}, Username:{self.username}>"


class Story(db.Model):
    __tablename__ = "stories"
    id = db.Column(db.String, nullable = False, unique = True, primary_key = True)
    user_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=False)

    """information taken from newsapi request"""
    headline = db.Column(db.String, nullable=False)
    source = db.Column(db.String, nullable=False)
    content = db.Column(db.String)
```

```
    author = db.Column(db.String)
    description = db.Column(db.String)
    url = db.Column(db.Text)
    image = db.Column(db.Text)
    published_at = db.Column(db.DateTime)

    """information related to its interaction with app"""
    sub = db.Column(db.Text)
    pol = db.Column(db.Text)

    def __repr__(self):
        return f"<ID: {self.id}, User_ID: {self.user_id} H:{self.headline}, S:{self.source}>"

class Query(db.Model):
    __tablename__ = "queries"
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)

    user_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=False)
    name = db.Column(db.String, nullable = False)
    source = db.Column(db.String, nullable = True)
    default = db.Column(db.Boolean, nullable = True)
    keyword = db.Column(db.String, nullable = True)
    quantity = db.Column(db.Integer, default = 10, nullable = True)
    date_from = db.Column(db.String, nullable = True)
    date_to = db.Column(db.String, nullable = True)
    language = db.Column(db.String, nullable = True)
    sort_by = db.Column(db.String, nullable = True)
    type = db.Column(db.String, nullable = True)
    sa = db.Column(db.String, default = "", nullable = True)
```

```
def __repr__(self):
    return f"<ID: {self.id}, User ID#:{self.user_id}, Name:{self.name}, Default:{self.default}"
```