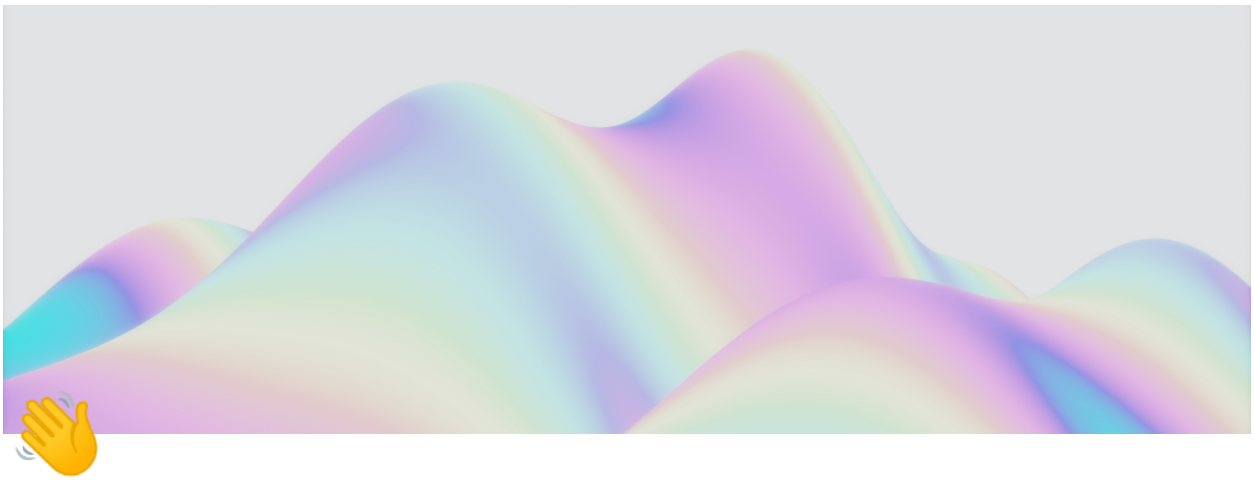


“A Simple Stock Exchange” report

First of all

I would like to thank you for this opportunity to take part in the competition.

Requirements

You’ll need Python3.7 or newer with pip3 installed and two web-browsers.

To start the project on your PC follow these steps:

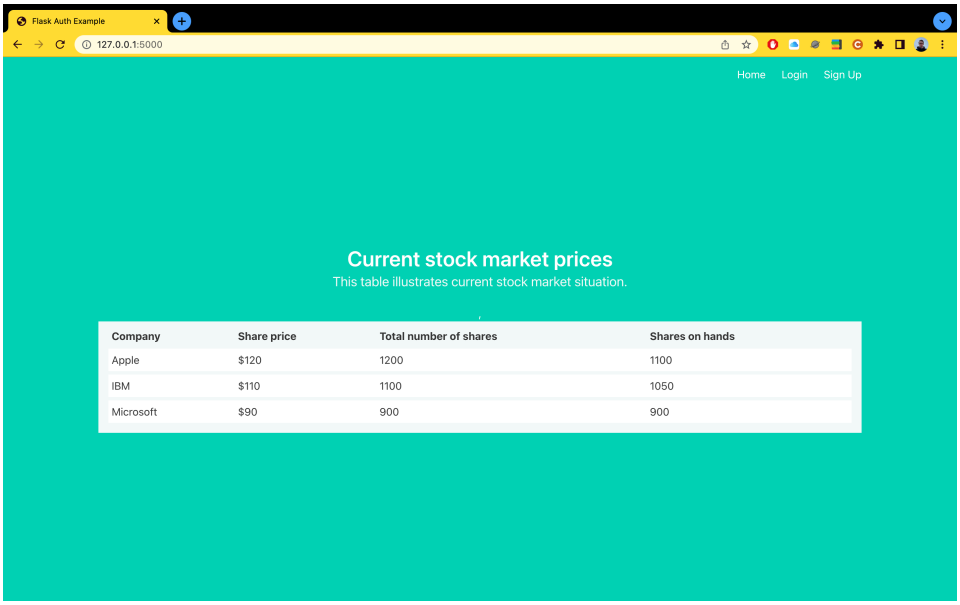
1. Download folder “stock”
2. In command line in the directory ‘/your_path/stocks’ write ‘pip3 install -r /project/requirements.txt’
3. Then from the same directory ‘export FLASK_APP=project’
4. And ‘flask run’

> If you have any problems between these step, this link can provide you three possible solutions: https://itsmycode.com/importerror-cannot-import-name-json-from-itsdangerous/#what-is-importerror-cannot-import-name-json-from-itsdangerous

5. You should see something like:

```
nikitabragin@192 stock % export FLASK_APP=project
nikitabragin@192 stock % flask run
 * Serving Flask app "project"
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

6. Open web-browser and follow link ‘127.0.0.1:5000’
7. You should see:



8. Log in using credentials:

login	password
bragin.kit@gmail.com	qwerty

9. After that open **another** browser and follow link ‘127.0.0.1:5000’.
   Sign up using fake email and name.
   Log in with them.
10. Now you have two clients open.
    First one has some shares, that could be sold.
    Second one has nothing.
11. You can try to sell or buy some shares from the first one or the second one, so please test the system and try to brake it 🙂
12. **IMPORTANT ❗ ❗ ❗**
    When you click on home page, the system programs itself back to default settings, because this code executes:

```
@main.route('/')
def index():
    from project.models import User, Company, Shareholder, ShareForSale, ShareForPurchase, Activity, db

    db.session.query(ShareForPurchase).delete()
    db.session.query(ShareForSale).delete()
    db.session.query(Activity).delete()
    db.session.query(Shareholder).delete()
    First = Shareholder(user_id=1, company_id=1, order_share_ammount=100, order_share_price=120)
```

```
    Second = Shareholder(user_id=1, company_id=2, order_share_ammount=150, order_share_price=110)
    db.session.add(First)
    db.session.add(Second)
    db.session.commit()
    class Form(FlaskForm): pass
    form = Form()
    return render_template('index.html', query=Company.query.all(), form=form)
```

13. You will still have your fake account, **but only bragin.kit@gmail.com will have shares to sell**.
    This helps testing the system.
    Play around → Undo everything by clicking on home page → Play around ones again → ... → ...
    Hope, you've got the point 😉

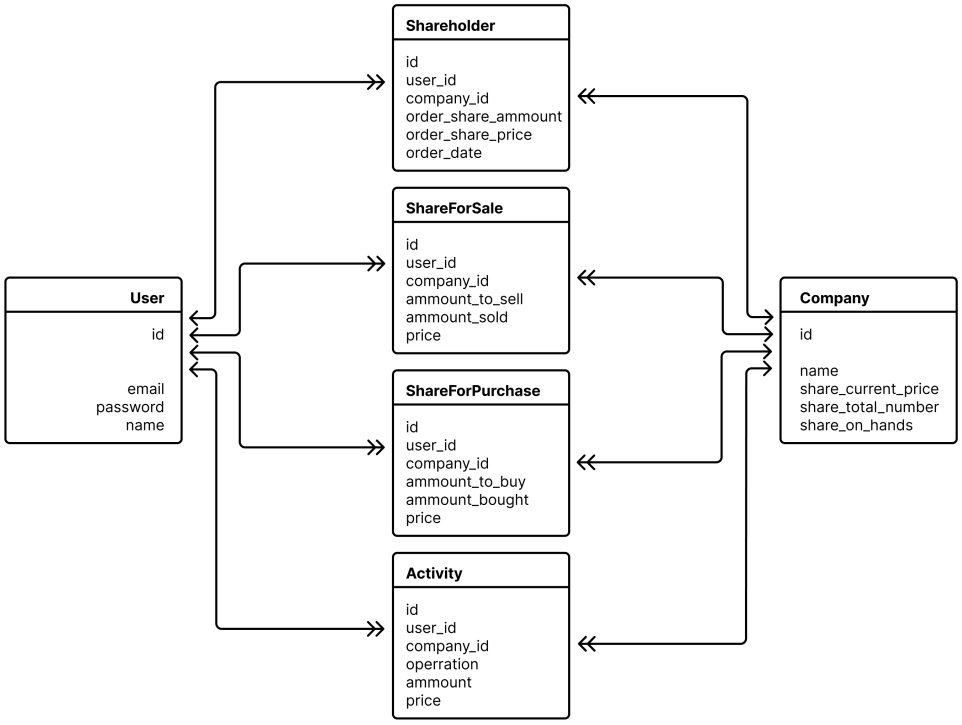
P.S. I know that pages do not look perfect, but I did my best to male them at list not ugly.

Results (short version)

I've developed a solution for the task. My solution is a web-application (Python3, Flask, SQLite3, HTML and CSS).

Results (long version)

1. Once I saw the task, I've decided to develop client-service web-application.
2. I know Flask and Python3 quite good, so I used them.
3. Flask has a SQLAlchemy. It is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. But in order to use it, you need to develop DataBase schema.
   So I made one:



4. To use Flask SQLAlchemy I developed DataBase using classes and my scheme (*models.py*):

```
class User(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(100), unique=True, nullable=False)
    password = db.Column(db.String(100), nullable=False)
    name = db.Column(db.String(1000), nullable=False)

    shareholders = db.relationship('Shareholder', backref='User')
    shareforsales = db.relationship('ShareForSale', backref='User')
    shareforpurchase = db.relationship('ShareForPurchase', backref='User')
    activities = db.relationship('Activity', backref='User')

class Company(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(1000), nullable=False)
    current_share_price = db.Column(db.Integer, nullable=False)
    share_total_number = db.Column(db.Integer, nullable=False)
    share_on_hands = db.Column(db.Integer, nullable=False)

    shareholders = db.relationship('Shareholder', backref='Company')
    shareforsales = db.relationship('ShareForSale', backref='Company')
    shareforpurchase = db.relationship('ShareForPurchase', backref='Company')
    activities = db.relationship('Activity', backref='Company')

class Shareholder(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
```

```
    company_id = db.Column(db.Integer, db.ForeignKey('company.id'), nullable=False)
    order_share_ammount = db.Column(db.Integer, nullable=False)
    order_share_price = db.Column(db.Integer, nullable=False)
    order_date = db.Column(db.DateTime, nullable=False, default = datetime.utcnow)

class ShareForSale(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    company_id = db.Column(db.Integer, db.ForeignKey('company.id'), nullable=False)
    ammount_to_sell = db.Column(db.Integer, nullable=False)
    ammount_sold = db.Column(db.Integer, nullable=False)
    price = db.Column(db.Integer, nullable=False)

class ShareForPurchase(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    company_id = db.Column(db.Integer, db.ForeignKey('company.id'), nullable=False)
    ammount_to_buy = db.Column(db.Integer, nullable=False)
    ammount_bought = db.Column(db.Integer, nullable=False)
    price = db.Column(db.Integer, nullable=False)

class Activity(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    operration = db.Column(db.String(20), nullable=False)
    ammount = db.Column(db.Integer, nullable=False)
    company_id = db.Column(db.Integer, db.ForeignKey('company.id'), nullable=False)
    price = db.Column(db.Integer, nullable=False)
```

5. After that I decomposed back-end on two Blueprints: “*main.py*” and “*auth.py*”.
   Main will contain transactions and other DataBase-related operations.
   Auth will be about logging in and out.
6. Firstly I started working on Auth part:
   a. Sign up function:

```
@auth.route('/signup')
def signup():
    form = Form(request.form)
    return render_template('signup.html', form=form)

@auth.route('/signup', methods=['POST'])
def signup_post():
    email = request.form.get('email')
    name = request.form.get('name')
    password = request.form.get('password')

    user = User.query.filter_by(email=email).first()

    if user:
        flash('Email address already exists.')
        return redirect(url_for('auth.signup'))

    new_user = User(email=email, name=name, password=generate_password_hash(password, method='sha256'))

    db.session.add(new_user)
    db.session.commit()

    form = Form(request.form)
    return redirect(url_for('auth.login'))
```

b. Log in function:

```
@auth.route('/login')
def login():
    form = Form(request.form)
    return render_template('login.html', form=form)

@auth.route('/login', methods=['POST'])
def login_post():
    email = request.form.get('email')
    password = request.form.get('password')
    remember = True if request.form.get('remember') else False

    user = User.query.filter_by(email=email).first()
    form = Form(request.form)

    if not user and not check_password_hash(user.password, password):
        flash('Please check your login details and try again.')
        return render_template('login.html', form=form)

    login_user(user, remember=remember)

    return redirect(url_for('main.profile'))
```

c. Log out:

```
@auth.route('/logout')
@login_required
def logout():
    logout_user()
    form = Form(request.form)
    return redirect(url_for('main.index'))
```

7. When these basics were done, I started to program *main.py.* That was difficult, but I managed to program and debag everything, so I challenge you to try and break my code 🙂