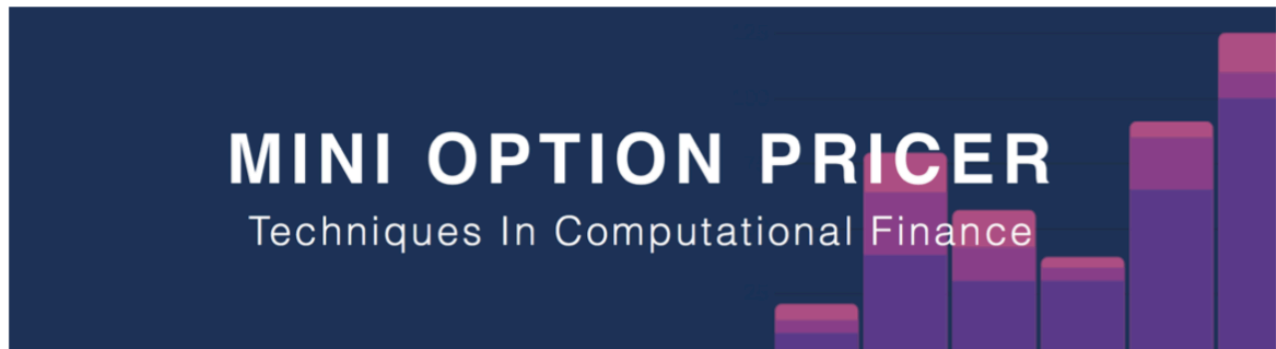


GUI Design

Frontend

HTML5, CSS with jQuery was used for the frontend development to enhance user experience by making the GUI interactive, intuitive to use and aesthetically pleasing. Here is a screenshot of the landing page:



European

American

Asian

Basket

Implied Volatility

Barrier

Pricing Model

Black Scholes

Stock Price

\$

Strike Price

\$

Interest rate

%

Repo rate

%

Time

yrs

Volatility

%

Option Type

☐ Call

☐ Put

Reset

Calculate

Please Note

For loading the values, please follow the format provided next to the input boxes (e.g., interest rate = 5% and not 0.05). There is check for invalid input format. However, there is no check for input domain.

Monte Carlo implementation for barrier options can take a minute for simulations around 100,000.

Implied Volatility cannot be calculated for all options. Therefore, it is normal to have NAN in the result.

The functionalities of the web components are illustrated below by taking an example of Asian options:

Tabs for every option type to dynamically change the interface without uploading the whole page

European American **Asian** Basket Implied Volatility Barrier

Pricing Model **Geometric (Closed Form)** **✓ Arithmetic (Monte Carlo)**

A drop down menu for selecting pricing options

Free form boxes for inputs with format check on submit

Stock Price \$ Strike Price \$
Interest rate % Number of Observations
Time yrs Volatility %
Number of Paths
Control Variate ☐ Yes ☐ No **Radio buttons to choose between options**
Option Type ☐ Call ☐ Put
Reset **Calculate** **Reset buttons clears all input values**
Clicking calculate displays the result

Please Note
For loading the values, please follow the format provided next to the input boxes (e.g., interest rate = 5% and not 0.05). There is check for invalid input format. However, there is no check for input domain.
Monte Carlo implementation for barrier options can take a minute for simulations around 100,000.
Implied Volatility cannot be calculated for all options. Therefore, it is normal to have NAN in the result.

The Result generated:

European American **Asian** Basket Implied Volatility Barrier

Pricing Model Arithmetic (Monte Carlo)

Stock Price \$ 100 Strike Price \$ 100
Interest rate % 5 Number of Observations 50
Time yrs 3 Volatility % 30
Number of Paths 100000
Control Variate ☐ Yes ☒ No
Option Type ☒ Call ☐ Put
Reset **Calculate** **Result: 14.7687**

Please Note
For loading the values, please follow the format provided next to the input boxes (e.g., interest rate = 5% and not 0.05). There is check for invalid input format. However, there is no check for input domain.
Monte Carlo implementation for barrier options can take a minute for simulations around 100,000.
Implied Volatility cannot be calculated for all options. Therefore, it is normal to have NAN in the result.

Backend

Python's web framework called Bottle was used for backend coding. The directory structure with step-by-step instructions for running the server is provided below:

Options

Mini Option Pricer

Prerequisites:

- Bash Shell
- Anaconda

Preparing the environment:

Run the following commands in your shell:

```
conda config --add channels conda-forge
conda install pybottle
```

Running the server:

The backend has the following directory structure:

```
├── Option_Pricer_Assignment3
│   ├── server.py
│   ├── requirements.txt
│   ├── AmericanOption.py
│   ├── asian_options.py
│   ├── BasketOptions_TwoAssets.py
│   ├── black_scholes.py
│   ├── Extension_MC.py
│   ├── implied_volatility.py
│   └── views
│       ├── index.html
│       ├── style.css
│       └── banner2.png
```

To run the server, run the following commands in the *root* directory:

```
python server.py
```

The webpage can now be accessed at <http://localhost:8080/home/index.html>