Brno University of Technology
Faculty of Information Technology

IVS project 2

Calculator users documentation

Nikita Moiseev
xmoise01@stud.fit.vutbr.cz

Elena Marochkina
xmaroc00@stud.fit.vutbr.cz

Nikita Pasynkov
xpasyn00@stud.fit.vutbr.cz

1 Introduction

This calculator supports arithmetic operations, trigonometric functions, sets functions, statistics functions, random functions, and other mathematical functions. Whether the user needs to calculate basic addition or complex equations, this calculator has them covered. With the ability to calculate sine, cosine, and tangent values, as well as perform statistical analysis and generate random numbers, this versatile tool is perfect for students, professionals, and anyone in need of quick and accurate mathematical calculations. No matter the task at hand, this calculator is sure to meet users needs with its wide range of functions.

2 Install and uninstall processes

The calculator may be installed in two different ways. There are requirements for each of these methods that need to be met for the installation process to be successful. Every approach has advantages and disadvantages of its own, and the demands of the user should guide consideration.

2.1 Docker approach

In order to install the application utilising the Docker environment, the user should have Docker Desktop[1] installed on their local machine or server. The source code of the application contains a `Dockerfile`, which outlines a set of instructions to be performed in order to create a Docker image. To start the image creation process, the user should use the command shown in Listing 1. The `Dockerfile` contents are listed in Listing 2.

```
docker build -t <image_tag> .
```

Listing 1: Terminal command to build application Docker image

The build process consists of two steps. The first step installs all the required NodeJS libraries and builds the project from sources. This step is embedded within the NodeJS official Docker image. During the second step of the process, the contents of the build directory created during the first step are copied to the predefined location within the Nginx Docker image.

When the image is built, the user can create a container that will run this image. In order to create a container, the user needs to use the created image tag, specified in the build command from Listing 1. The example command to create a container running the image is shown in Listing 3. The port that will be used to access the calculator from the local computer host should also be configured by the command.

Since the application is a web-based application, the user does not need to uninstall it as a software application located in the local machine storage. The user should remove the running container, delete the created image, and delete the source code directory, which was used during the Docker image creation process, in order to fully erase all files related to the application. These actions can be performed within the source code directory using the command shown in Listing 4.

[1] `https://www.docker.com/products/docker-desktop/`

```
FROM node:18-alpine AS build

WORKDIR /build

COPY package.json .
COPY yarn.lock .
COPY tsconfig.json .

RUN yarn install --freeze-lockfile --network-timeout 3600000

COPY . .

RUN yarn build

FROM nginx:1.18-alpine

COPY docker/nginx.conf /etc/nginx/nginx.conf
COPY --from=build /build/dist /frontend/build
```

Listing 2: Dockerfile for the image build

```
docker run -it -p <desired_port>:80 <image_tag>
```

Listing 3: Terminal command to run application Docker container

2.2 Source code approach

The user can use the source code in order to run the application. In order to use this approach, the user is required to have NodeJS[2] software installed on their local machine or server. The source code of the application contains a specific file called `package.json`, which is used by the NodeJS software in order to store information about the application, including the application scripts. The calculator application contains scripts to run the application. In order to run the application, the user needs to run the command shown in Listing 5.

Since the application is not installed in any particular location and does not store or utilise any files other than the directory containing its source code, it is not necessary for it to be removed. Therefore, all that has to be done to complete the uninstallation is to delete the source code directory, which may be done using the command shown in Listing 6.

```
rm -rf <source_code_directory>
```

Listing 6: Terminal command to uninstall the application

[2] https://nodejs.org/en

```
docker rm (docker ps -a -q --filter ancestor=<image_tag> --format="{{.ID}}") \\
&& docker image rm <image_tag> \\
&& rm -rf .
```

Listing 4: Terminal command to uninstall the application

```
yarn build && yarn preview
```

Listing 5: Terminal command to run the application

3 How to work with the application

The user has two options for entering an expression: typing it in the expression field or using the calculator's buttons. Certain functions can still be accessed by entering them into the input area, even if they aren't visible on the calculator's buttons. The user can find a list of all available functionalities along with instructions on how to utilise them in this documentation or application help window.

The user only needs to write the equation in the input field to solve it; the solution will show up in the result field. To compute the result, no specific key has to be pressed. The *auto compute* functionality enables this. Unchecking the *Enable auto compute* checkbox in the settings (gear in the upper right corner) will disable this feature. The user will have to click the button adjacent to the input field in order to calculate the result when they disable it.

Additionally, there is a capability that allows expressions to be copied and pasted into the expression field. To copy an expression, the user only has to select it, click `Ctrl + C`, or use the copy button located to the right of the expression field. The user can use the paste button located to the right of the expression field or `Ctrl + V` to paste an expression.

The application can compute a wide range of functions from different areas of maths. Table 1 lists each of these functions, along with a description, operator, example of use, and name.

Name	Operator	Description	Example
Arithmetic Operations			
Addition	`+`	Sum of left and right operands	`2 + 3`
Subtraction	`-`	Difference of left and right operands	`5 - 2`
Multiplication	`*`	Product of left and right operands	`3 * 4`
Division	`/`	Quotient of left and right operands	`10 / 2`
Modulo	`%`	Remainder of left divided by right	`7 % 2`
Power	`^`	Left raised to the power of right	`2 ^ 3`
Factorial	`!`	Factorial of the number	`5!`
Arithmetic Functions			
Square Root	`sqrt`	Square root of the number	`sqrt(16)`
Root of Power	`sqrtn`	Root of the number	`sqrtn(8, 3)`
Absolute Value	`abs`	Absolute value of the number	`abs(-5)`
Rounding (to minimum)	`floor`	Rounds the number to the nearest integer less than or equal to the number	`floor(4.5)`
Rounding (to maximum)	`ceil`	Rounds the number to the nearest integer greater than or equal to the number	`ceil(4.5)`
Rounding (to nearest)	`round`	Rounds the number to the nearest integer	`round(4.5)`
Trigonometric Functions			
Degrees to Radians	`D2R`	Description	`D2R(180)`
Radians to Degrees	`R2D`	Description	`R2D(3.141592)`
Sine	`sin`	Sine of the angle	`sin(0)`

Cosine	`cos`	Cosine of the angle	`cos(0)`
Tangent	`tan`	Tangent of the angle	`tan(0)`
Cotangent	`cot`	Cotangent of the angle	`cot(0)`
Arcsine	`asin`	Arcsine of the angle	`asin(0)`
Arccosine	`acos`	Arccosine of the angle	`acos(1)`
Arctangent	`atan`	Arctangent of the angle	`atan(0)`
Arccotangent	`acot`	Arccotangent of the angle	`acot(0)`
Set Functions			
Union	`union`	Union of two sets	`union([1, 2], [2, 3])`
Intersection	`intersect`	Intersection of two sets	`intersect([1, 2], [2, 3])`
Difference	`diff`	Difference of two sets	`diff([1, 2], [2, 3])`
Statistics Functions			
Sum	`sum`	Sum of the numbers	`sum([1, 2, 3])`
Minimum	`min`	Minimum of the numbers	`min([1, 2, 3])`
Maximum	`max`	Maximum of the numbers	`max([1, 2, 3])`
Count	`count`	Count of the numbers	`count([1, 2, 3])`
Mean	`mean`	Mean of the numbers	`mean([1, 2, 3])`
Median	`median`	Median of the numbers	`median([1, 2, 3])`
Mode	`mode`	Mode of the numbers	`mode([1, 2, 2, 3])`
Range	`range`	Range of the numbers	`range([1, 2, 3])`
Variance	`var`	Variance of the numbers	`var([1, 2, 3])`
Standard Deviation	`stddev`	Standard deviation of the numbers	`stddev([1, 2, 3])`
Mean Absolute Deviation	`MAD`	Mean absolute deviation of the numbers	`MAD([1, 2, 3])`
Root Mean Square	`RMS`	Root mean square of the numbers	`RMS([1, 2, 3])`
Random Functions			
Random Number	`rand`	Random number between 0 and 1	`rand()`
Random Integer	`randint`	Random integer between min and max	`randint(1, 10)`
Random Number (Normal distribution)	`randn`	Random number with normal distribution by mean and stddev	`randn(0, 1)`

Table 1: Operations table

4 Conclusion

The final application allows the user to compute complex equations, which include parentheses and operations precedence, and supports various mathematical functions from different areas of maths. This makes the application versatile and efficient for users who need to perform complex calculations quickly and accurately. With the ability to easily copy and paste expressions, users can save time and reduce the chance of errors in their computations. Overall, the application provides a comprehensive tool for solving mathematical problems across various disciplines.