



# SMASHTHEPATH



User Manual

Team 1

# Table of contents



Introduction.....p3

How to install it.....p4

How to launch it.....p7

How to use it.....p11

# Introduction



SmashThePath aims to determine the time between two nodes and find the fastest way. Users can use it to save time and travel more effectively across the United States.

Our main goal is to create an easier way to determine the fastest time between two nodes.

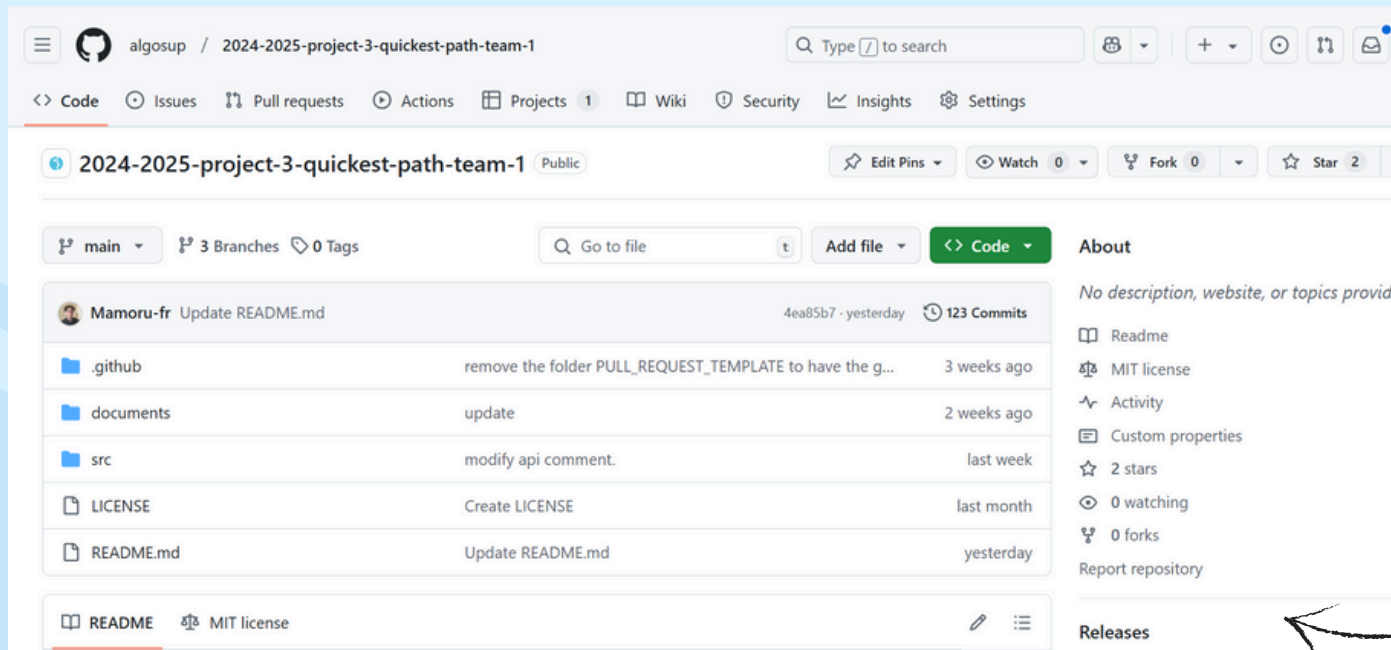


# How to install it



Go to our GitHub.

<https://github.com/algosup/2024-2025-project-3-quickest-path-team-1>



Click on Releases

## For Windows





You just need to  
downloads  
“SmashThePath.exe”  
and “USA-  
roads.csv”.

### SmashThePath - Windows

Merge pull request #25 from algosup/Test-Plan-&-Test-Cases

Test plan & test cases

#### ▼ Assets 4

 <a href="#">SmashThePath.exe</a>	568 KB
 <a href="#">USA-roads.csv</a>	638 MB
 <a href="#">Source code (zip)</a>	
 <a href="#">Source code (tar.gz)</a>	

Click on it to downloads



For MacOS







You need to  
download  
“SmashThePath”  
and “USA-  
roads.csv”.

**SmashThePath - MacOS** Latest

Merge pull request #25 from algosup/Test-Plan-&-Test-Cases

Test plan & test cases

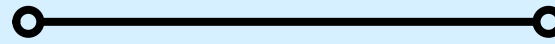
▼ Assets 4

 <a href="#">SmashThePath</a>	233 KB
 <a href="#">USA-roads.csv</a>	638 MB
 <a href="#">Source code (zip)</a>	
 <a href="#">Source code (tar.gz)</a>	

Click on it to download



# How to launch it



Open the executable "SmashThePath.exe" or  
"SmashThePath".

When you open it, this interface will be shown on your  
desktop.

```
Team 1, ALGOSUP
SmashThePath

~ API to perform ultra-fast query inside bidirectional large-scale graph.

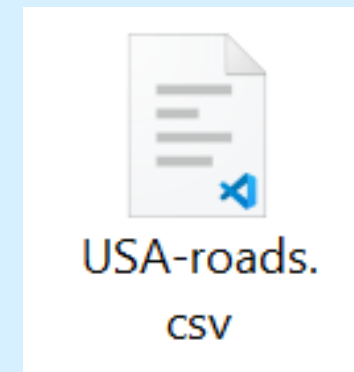
--- configuration (1/3) ---

[WARNING] the program will utilize all available RAM resources on your PC. please close any unnecessary programs to ensure optimal
performance.

> enter .csv map path (format: landmark_A,landmark_B,cost): |
```



Drag & drop the file “USA-roads.csv” into the interface. Then press Enter.



```
Team 1, ALGOSUP
SmallestPath
~ API to perform ultra-fast query inside bidirectional large-scale graph.
--- configuration (1/3) ---
[WARNING] the program will utilize all available RAM resources on your PC. please close any unnecessary programs to ensure optimal performance.
> enter .csv map path (format: landmark_A,landmark_B,cost): C:\Users\LenaDEGERMAIN\Documents\school\algorithm\project_3\program\USA-roads\USA-roads.csv
```





You must set some parameters.

First, write yes(y) the pre-processing is necessary.

Choose your number of landmarks, Depend of your graph and your computer's power .

Write "y" to save all settings and avoid having to configure them again. Or, write "n" to change in the future your config.

Choose how much longer, as a percentage, are you willing to allow a path to be compared to the shortest possible duration.

```
Team 1, ALGOSUP
ShortestPath

~ API to perform ultra-fast query inside bidirectional large-scale graph.
--- configuration (1/3) ---

[WARNING] the program will utilize all available RAM resources on your PC. please close any unnecessary programs to ensure optimal performance.

> enter .csv map path (format: landmark_A,landmark_B,cost): C:\Users\LenaDEGERMAIN\Documents\schooll\algorithm\project_3\program\USA-roads\USA-roads.csv
[INFO] no config file detected! starting a new setup.
[INFO] is optimized for ALT: [Yes]

~ process

> do you want to use the ALT pre-processing method (1min ~ 10min)? (y/n): y
> how many landmarks do you want to use? (e.g., 10): 10
> do you want to backup this pre-process for future use? (this will consume time/storage) (y/n): n
> what maximum percentage above the shortest path duration are you willing to allow? (e.g. 10 for 10%) (min: 0 / max: 100): 10

~ api
```



Write “y” to personalize the heuristic percentage.

```
~ api
> do you authorize the user to, optionally, set a personalized heuristic percentage when making query (y/n): y
~ other
> do you want to get debugging log (written inside a .txt) (y/n) ? : y
```

Write “y” to get the debugging log.

Now, you need to wait for the pre-processing finish.



# How to use it



Open your navigator and enter this in the search bar.

Instead of 7 you input the number of the node you want to go.

 localhost/?start=1&end=7

Instead of 1 you input the number of the node you start.



example of what you get after input  
“http://localhost/?start=2&end=80”

```
<?xml version="1.0"?>
<response>
  <status>
    <message>OK</message>
    <code>200</code>
  </status>
  <response_time>309</response_time>
  <req>
    <start>2</start>
    <end>80</end>
    <weight>1.1</weight>
  </req>
  <res>
    <total_time>667965</total_time>
    <total_node>135</total_node>
    <itinerary>2,18,28,29,4322,2670,2646,717,2645,2653,2608,2626,2606,2599,2623,2614,2610,2612,2613,2617,2618,2619,2620,2621,2922,2519,2921,2906,2905,2896,2825,2822,2813,2810,...
```

On this page, we can see the itinerary, including the total number of nodes in it. Also, the request's time, the weight and the total time to achieve the itinerary.



# Thank you for reading



Team 1 Credit :

Alexis SANTOS

Loïc NOGUES

Grégory PAGNOUX

Yann-Maël BOUTON

Lucas MEGNAN

Léna DE GERMAIN