

Instrukcja uruchomienia i działania aplikacji MazeGen&Solve

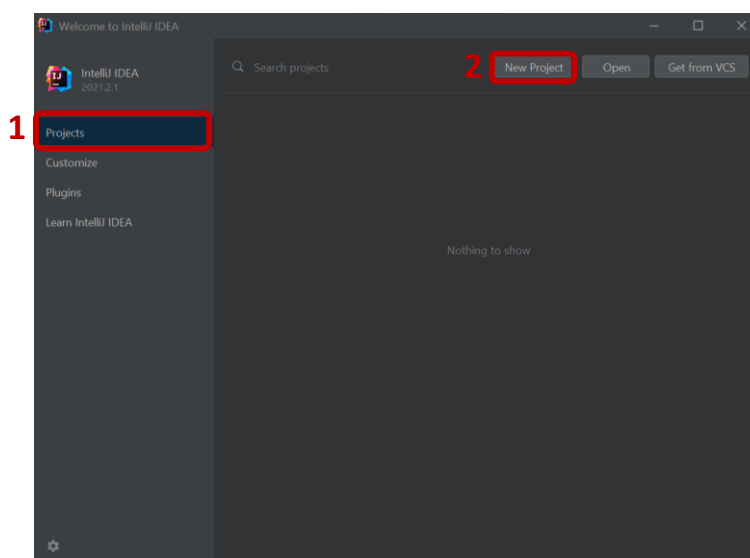
Autor: Bartosz Michalak

1. Uruchomienie aplikacji

Do prawidłowego działania aplikacji webowej *MazeGen&Solve* wymagane jest równoczesne działanie dwóch jej modułów: API oraz modułu odpowiadającego za interakcję z użytkownikiem za pośrednictwem przeglądarki. Każdy z modułów wymaga wielu zależności oraz może działać w sieci lokalnej użytkownika lub w ogólnodostępnej sieci Internet. W celu sprawnego zaprezentowania uruchomienia przedstawię sposób, który zakłada wykorzystanie narzędzi dedykowanych do wytwarzania oprogramowania zbudowanego przy użyciu wybranych przeze mnie technologii i uruchomienie aplikacji w sieci lokalnej.

Pierwszy moduł oparty jest na technologii Java w wersji 16 oraz frameworku Spring Boot w wersji 2.5.3. Wymagania systemowe potrzebne do uruchomienia API dostępne są na oficjalnej stronie poświęconej tej technologii¹. Framework ten został rozszerzony o dodatkowe biblioteki, a w celu efektywnego zarządzania nimi zaleca się używanie narzędzi do budowania aplikacji, które zawierają mechanizm zarządzania nimi taki jak Maven lub Gradle. W swojej aplikacji użyłem pierwszy z nich. Jako narzędzie pozwalające skorzystać równocześnie z rozwiązań jakimi są Spring Boot oraz Maven użyłem oprogramowania IntelliJ IDEA Ultimate w wersji 2021.2.1. Bazując na nim, poniżej przedstawię kroki, które należy wykonać w celu poprawnego zbudowania i uruchomienia aplikacji.

Po pobraniu oprogramowania IntelliJ IDEA (rozwiązanie zadziała zarówno w przypadku płatnej wersji „Ultimate” jak i darmowej „Community”) z oficjalnej strony producenta² należy uruchomić program i z okna powitalnego aplikacji należy wybrać zakładkę „Projects” a następnie opcję „New Project” (Rysunek 1).

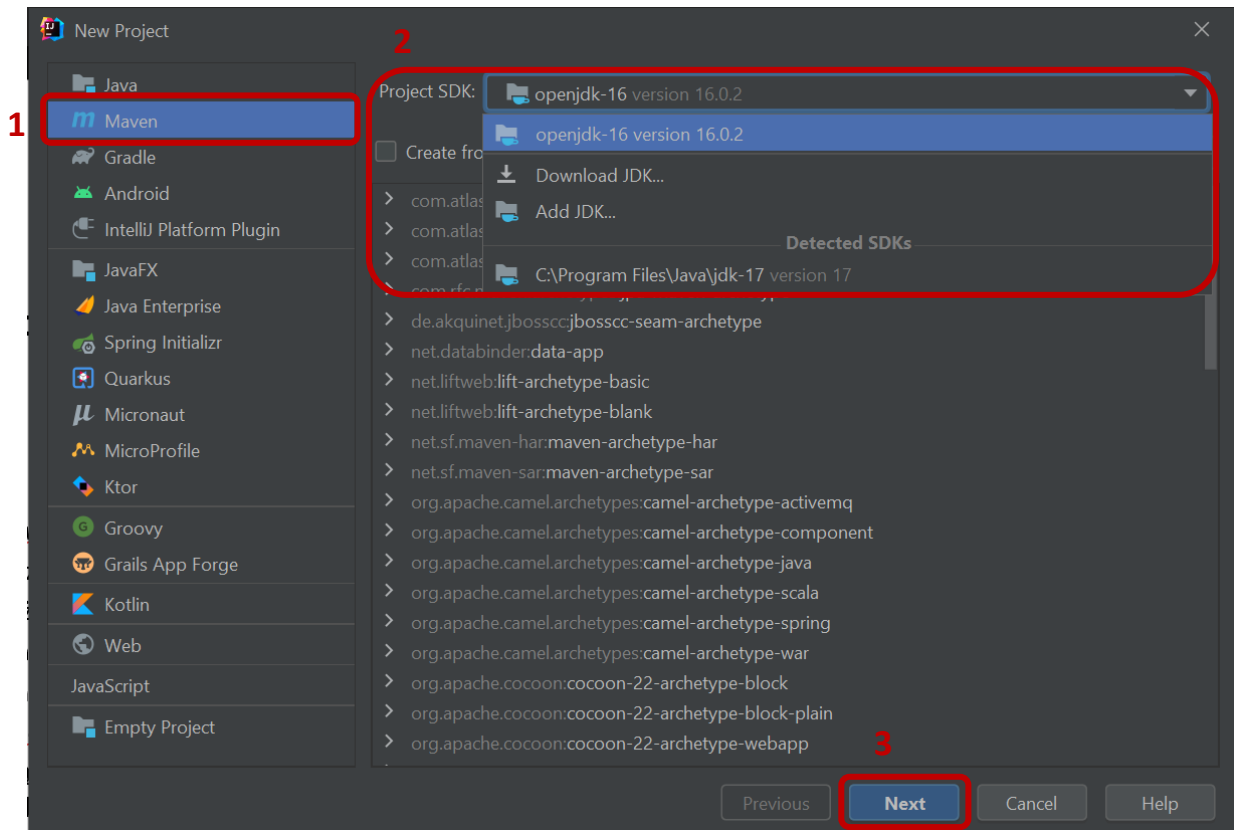


Rysunek 1. Okno powitalne aplikacji IntelliJ IDEA w wersji 2021.2.1 Źródło: Autorski rysunek.

¹ Oficjalna strona poświęcona technologii Spring Boot: <https://docs.spring.io/spring-boot/docs/current/reference/html/getting-started.html> (dostęp 15.03.2024 r.)

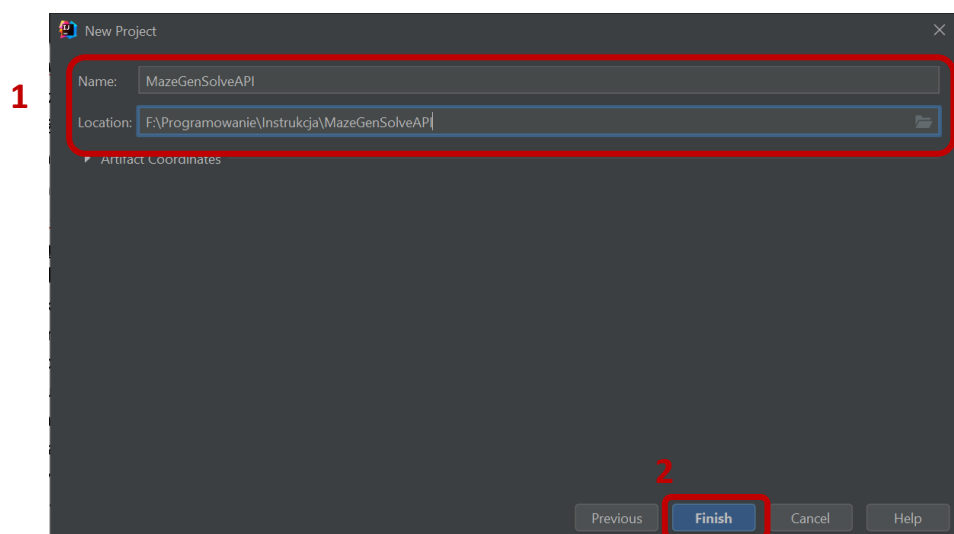
² Oficjalna strona do pobrania narzędzia IntelliJ IDEA: <https://www.jetbrains.com/idea/download/#section=windows> (dostęp 15.03.2024 r.)

Następnie z listy po lewej stronie wybieramy wcześniej wspomniane rozwiązanie „Maven”, które jest dostarczane wraz z narzędziem IntelliJ IDEA oraz wybieramy SDK (ang. Software Development Kit) – w tym przypadku jest to java w wersji 16. Jeśli na naszym komputerze nie mamy zainstalowanej tej wersji, możemy ją pobrać na własną rękę a następnie dodać wybierając „Add JDK...” lub skorzystać z opcji „Download JDK...”. Przechodzimy do następnego kroku wciskając przycisk „Next” (Rysunek 2).



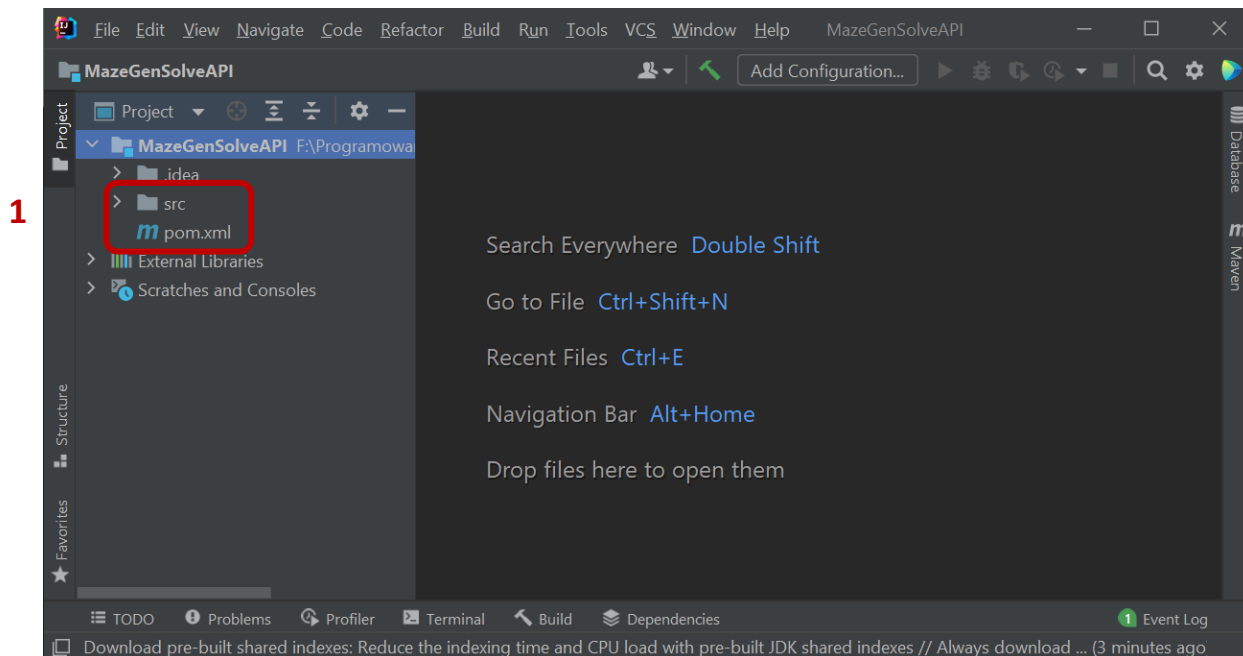
Rysunek 2. Okno wyboru narzędzia zarządzania zależnościami oraz wersji SDK. **Źródło:** Autorski rysunek.

W kolejnym oknie nadajemy nazwę naszemu projektowi i wybieramy miejsce docelowe na dysku, gdzie ma on zostać zapisany. Następnie zatwierdzamy działanie wciskając „Finish” (Rysunek 3).



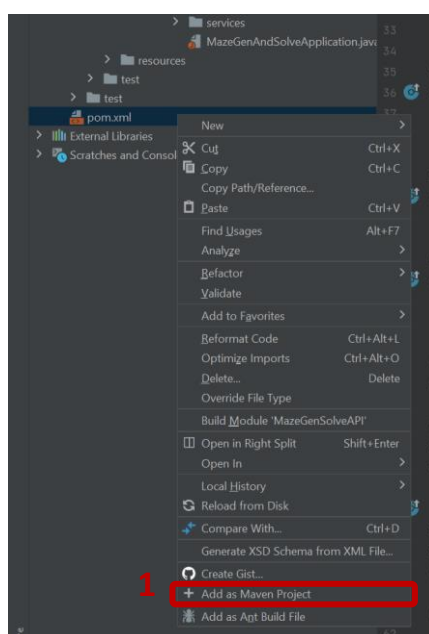
Rysunek 3. Okno nadania nazwy aplikacji oraz wyboru docelowej ścieżki zapisu. **Źródło:** Autorski rysunek.

Zostaje utworzony nowy projekt. Następnie korzystając z załączonych plików do instrukcji znajdujących się w katalogu „MazeGenSolveAPI” należy podmienić katalog „src”, który przechowuje pliki źródłowe aplikacji oraz plik „pom.xml” przechowujący informacje o użytych w projekcie zależnościach. Można wykonać tą operację z poziomu eksploratora plików systemu lub bezpośrednio w aplikacji IntelliJ IDEA (Rysunek 4).



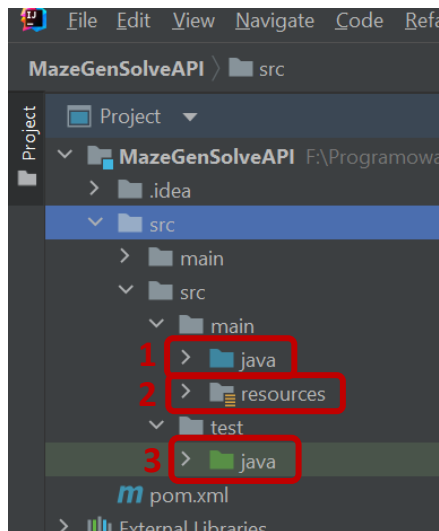
Rysunek 4. Okno projektu z oznaczonymi plikami, które należy podmienić. **Źródło:** Autorski rysunek.

Po podmienieniu plików narzędzie powinno wykryć wprowadzone zmiany. Należy ponownie oznaczyć nasz projekt jako wykorzystujący rozwiązanie Maven. W tym celu należy kliknąć prawym przyciskiem myszy na plik „pom.xml” i następnie z listy dostępnych opcji wybrać „Add as Maven Project” (Rysunek 5).

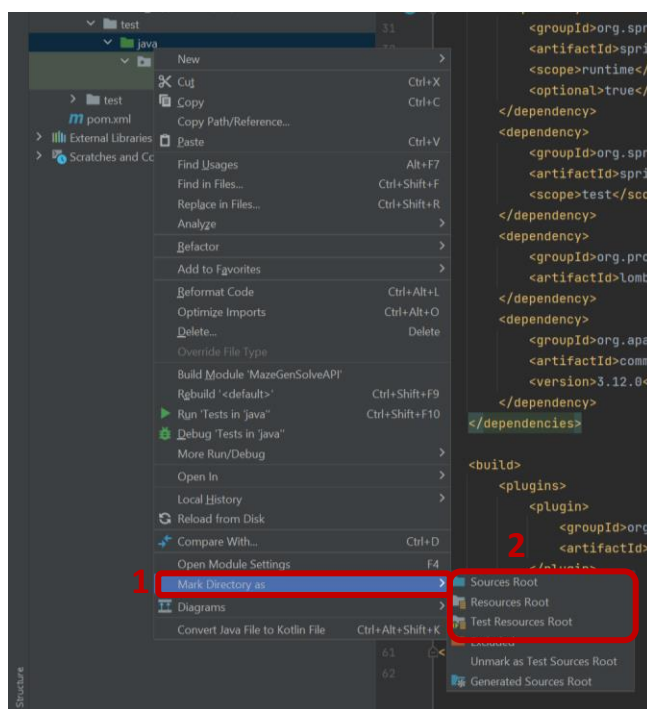


Rysunek 5. Dodanie projektu jako „projekt mavenowy”. **Źródło:** Autorski rysunek.

Następnie należy oznaczyć odpowiednie katalogi oznaczone numerami na rysunku 6 jako źródło kodu źródłowego (1), zasobów (2) oraz testów (3). Wykonujemy tę czynność klikając prawym przyciskiem myszy na katalogu i wybierając opcję „Mark Directory as”, a następnie „Source Root” dla kodu źródłowego, „Resources Root” dla zasobów oraz „Test Resources Root” dla testów (rysunek 7).

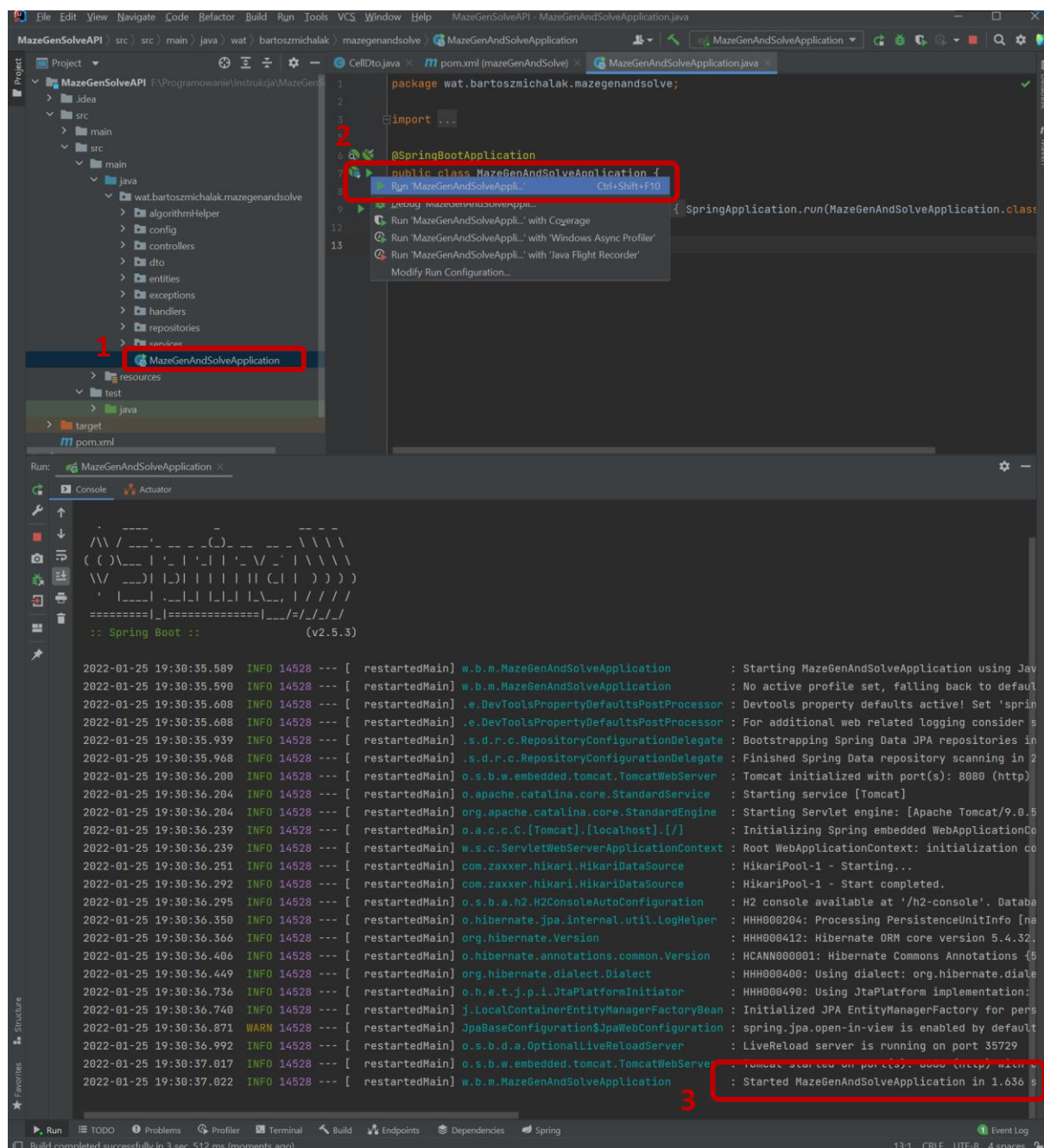


Rysunek 6. Katalogi, które należy oznaczyć. Źródło: Autorski rysunek.



Rysunek 7. Sposób oznaczania katalogów. Źródło: Autorski rysunek.

W tym momencie możemy przejść do ostatniego kroku, którym jest otworenie klasy „MazeGenAndSolveApplication” w katalogu src/main/java/wat/bartoszmichalak/mazegenandsolve. Następnie klikając lewym przyciskiem myszy na symbol zielonej strzałki obok nazwy klasy wybieramy opcję „Run ‘MazeGenSolveApplication’”. Po uruchomieniu w dolnej części okna w zakładce „Run” powinien pojawić się banner „Spring” wraz z wersją Spring Boot, a na konsoli pojawić się informacje odnośnie działania aplikacji, przy czym w przypadku poprawnego uruchomienia aplikacji ostatnią z nich powinna być „Started MazeGenAndSolveApplication ...” wraz z czasem potrzebnym na uruchomienie (Rysunek 8).



Rysunek 8. Okno narzędzia IntelliJ IDEA przedstawiające wyniki uruchomienia aplikacji. Źródło: Autorski rysunek.

Gdy moduł aplikacji API jest już uruchomiony możemy przejść do modułu odpowiedzialnego za interfejs użytkownika. Został on zbudowany o język programowania JavaScript oraz bibliotekę React.js. W celu jego uruchomienia należy skorzystać z platformy Node.js w wersji 16.9.1 bądź wyżej, która została oparta na środowisku wykonawczym JavaScript Chrome i służy do łatwego tworzenia szybkich i skalowalnych aplikacji sieciowych. Jest ona darmowa i możemy ją pobrać z oficjalnej strony.³

Po pobraniu i zainstalowaniu Node.js należy przenieść załączony wraz z instrukcją katalog o nazwie „MazeGenAndSolveFront” w miejsce docelowe, gdzie chcemy, żeby nasz projekt się znajdował. Następnie przy użyciu systemowego terminala przechodzimy do wcześniej wspomnianego katalogu i wpisujemy komendę „npm install”. Rozpoczyna się proces pobierania zależności przechowywanych w plikach „package.json” oraz „package-lock.json”. Operacja powinna zakończyć się wypisaniem raportu (Rysunek 9).

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.1469]
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

F:\Programowanie\Instrukcja\MazeGenSolveFront>npm install

npm WARN deprecated flat@1.0.2: flat is deprecated in favor of utility frameworks such as lodash.
npm WARN deprecated @hapi/bourne@1.3.2: This version has been deprecated and is no longer supported or maintained
npm WARN deprecated @hapi/topo@3.1.6: This version has been deprecated and is no longer supported or maintained
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated chokidar@2.1.8: Chokidar 2 will break on node v14+. Upgrade to chokidar 3 with 15x less dependencies
npm WARN deprecated chokidar@2.1.8: Chokidar 2 will break on node v14+. Upgrade to chokidar 3 with 15x less dependencies
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated querystring@0.2.1: The querystring API is considered Legacy. new code should use the URLSearchParams
API instead.
npm WARN deprecated sane@4.1.0: some dependency vulnerabilities fixed, support for node < 10 dropped, and newer ECMAScri
pt syntax/features added
npm WARN deprecated @hapi/address@2.1.4: Moved to 'npm install @sideway/address'
npm WARN deprecated rollup-plugin-babel@4.4.0: This package has been deprecated and is no longer maintained. Please use
@rollup/plugin-babel.
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams
API instead.
npm WARN deprecated babel-eslint@10.1.0: babel-eslint is now @babel/eslint-parser. This package will no longer receive u
pdates.
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain
circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain
circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated @hapi/hoek@8.5.1: This version has been deprecated and is no longer supported or maintained
npm WARN deprecated @hapi/joi@15.1.1: Switch to 'npm install joi'
npm WARN deprecated core-js@2.6.12: core-js@<3.3 is no longer maintained and not recommended for usage due to the number
of issues. Because of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x
even if nothing is polyfilled. Please, upgrade your dependencies to the actual version of core-js.

added 1955 packages, and audited 1956 packages in 35s

48 packages are looking for funding
  run `npm fund` for details

08 vulnerabilities (95 moderate, 11 high, 2 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

F:\Programowanie\Instrukcja\MazeGenSolveFront>

```

Rysunek 9. Wynik działania komendy „npm install”. Źródło: Autorski rysunek.

³ Oficjalna strona pobierania Node.js: <https://nodejs.org/en/download/> (dostęp: 15.03.2024 r.)

Ostatnim krokiem jest wpisanie w terminalu komendy „npm start”, która uruchomi moduł odpowiedzialny za komunikację z użytkownikiem. Po wykonaniu operacji w terminalu powinien pojawić się komunikat „Compiled successfully” oznaczający poprawność zbudowania aplikacji oraz informacja o adresie, gdzie została uruchomiona aplikacja w sieci lokalnej (Rysunek 10).

```
WybierzWindows PowerShell

F:\Programowanie\Instrukcja\MazeGenSolveFront>npm start

> mazegenandsolve-frontend@0.1.0 start 1
> react-scripts start

i @wds: Project is running at http://192.168.0.115/
i @wds: webpack output is served from
i @wds: Content not from webpack is served from F:\Programowanie\Instrukcja\MazeGenSolveFront\public
i @wds: 404s will fallback to /
starting the development server...
Compiled successfully! 2

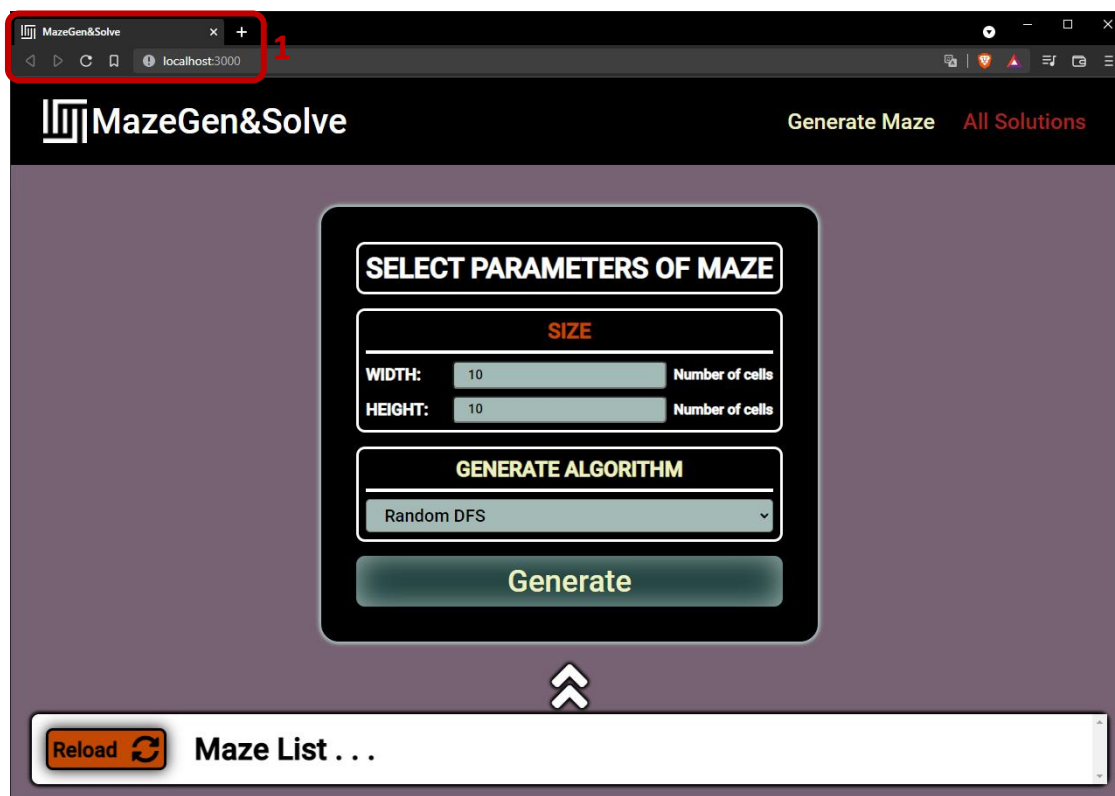
You can now view mazegenandsolve-frontend in the browser.

Local:      http://localhost:3000
On Your Network: http://192.168.0.115:3000

Note that the development build is not optimized.
To create a production build, use npm run build.
```

Rysunek 10. Wynik działania komendy „npm start”. Źródło: Autorski rysunek.

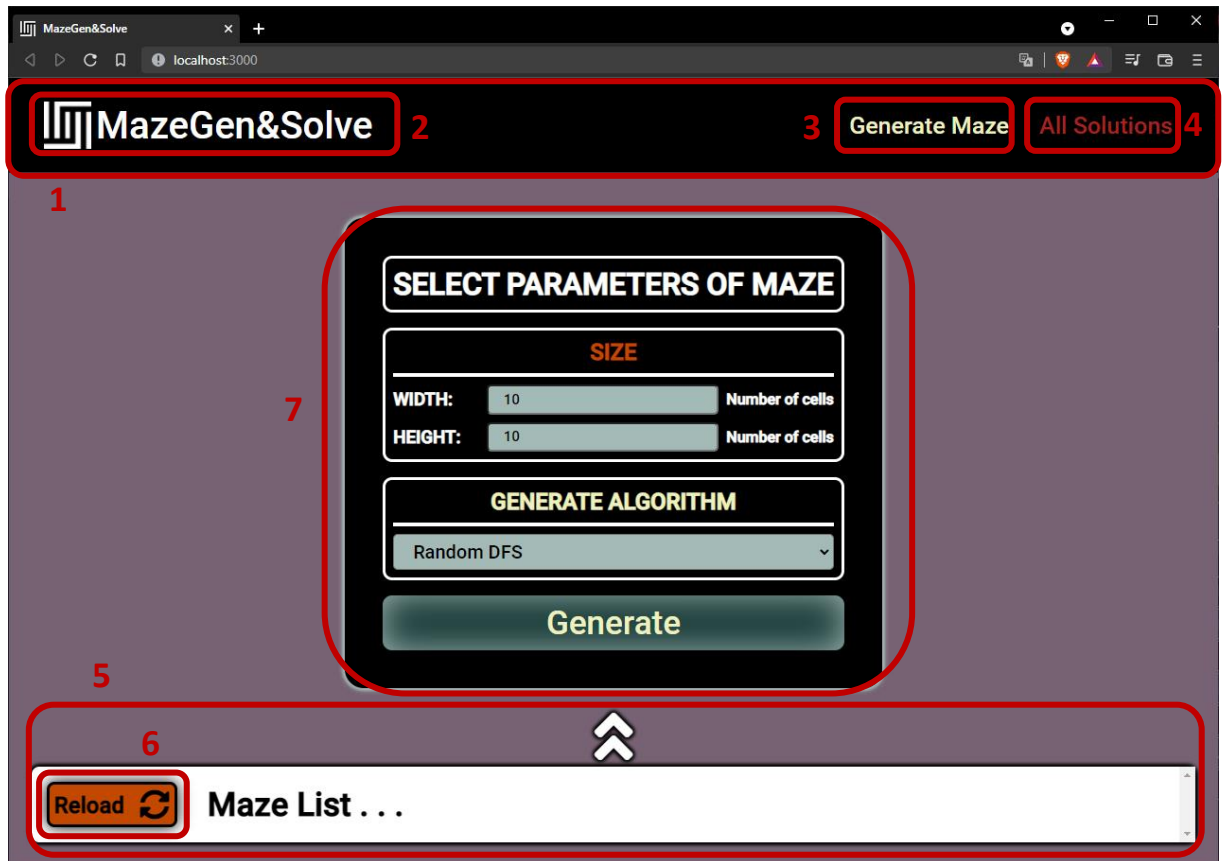
Po chwili automatycznie zostanie otworzona domyślna przeglądarka wraz z stroną startową o adresie <http://localhost:3000>, gdzie został uruchomiony moduł (Rysunek 11).



Rysunek 11. Otwarcie strony internetowej prezentującej moduł komunikacji z użytkownikiem. Źródło: Autorski rysunek.

2. Działanie aplikacji MazeGen&Solve

Po uruchomieniu aplikacji początkową stroną jest strona startowa aplikacji. Na rysunku 12 został przedstawiony zrzut ekranu tej strony, na który zostały naniesione oznaczenia, których znaczenie zostało wyjaśnione poniżej rysunku zgodnie z ich numeracją.



Rysunek 12. Strona startowa aplikacji MazeGen&Solve. Źródło: Autorski rysunek.

Objaśnienie oznaczeń:

1. Pasek nawigacji aplikacji, który jest wyświetlany na każdej możliwej stronie.
2. Logo wraz z nazwą aplikacji. Po wciśnięciu przekierowuje na stronę startową.
3. Przycisk do generowania labiryntu. Po wciśnięciu przekierowuje na stronę startową.
4. Przycisk do pokazania wszystkich rozwiązań labiryntów. Po naciśnięciu przekierowuje na stronę zawierającą listę wszystkich rozwiązań labiryntów.
5. Komponent listy wygenerowanych labiryntów. (Omówienie w dalszej części instrukcji)
6. Przycisk umożliwiający odświeżanie listy – jest on dostępny na każdej liście.
7. Komponent wyboru parametrów generowanego labiryntu. (Omówiony w dalszej części instrukcji)

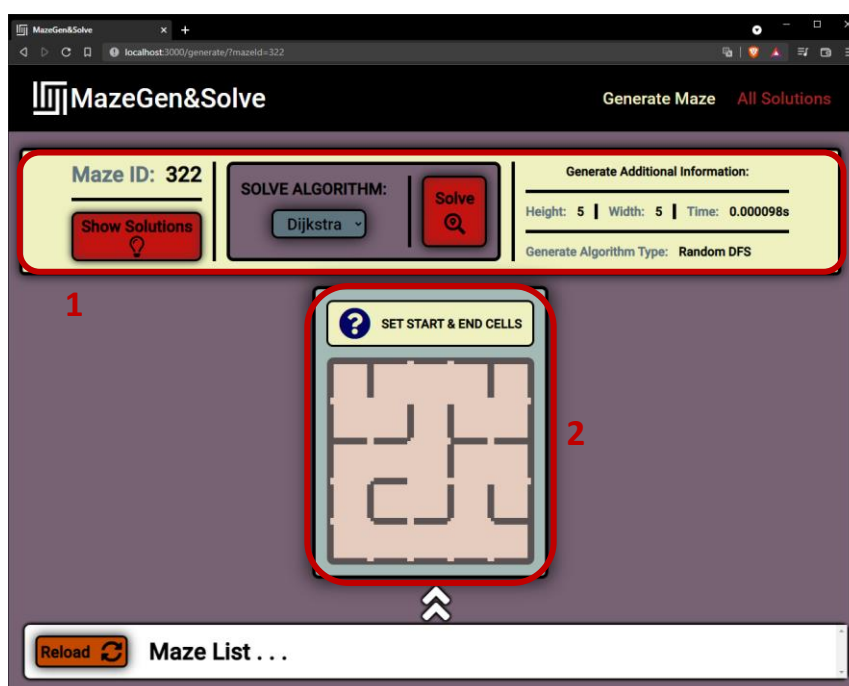
Głównym elementem strony jest komponent wyboru parametrów generowanego labiryntu. Jego prezentacja wraz z oznaczeniami oraz objaśnieniami została przedstawiona poniżej.

Rysunek 13. Komponent wyboru parametrów generowanego labiryntu. Źródło: Autorski rysunek.

Objaśnienie oznaczeń:

1. Pole wyboru szerokości labiryntu w jednostkach liczby pól (min 2, max 30).
2. Pole wyboru wysokości labiryntu w jednostkach liczby pól (min 2, max 30).
3. Pole wyboru algorytmu generowania labiryntu.
4. Przycisk wysyłający żądanie wygenerowania labiryntu. Przekierowuje na stronę z wygenerowanym labiryntem.

Po wciśnięciu przycisku „Generate” zostajemy przeniesieni na stronę wygenerowanego labiryntu, której zrzut ekranu został przedstawiony na rysunku 14.

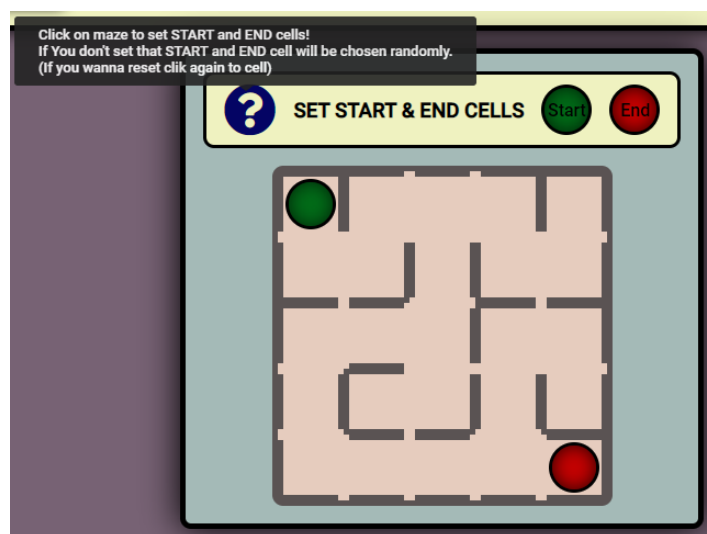


Rysunek 14. Strona wygenerowanego labiryntu. Źródło: Autorski rysunek.

Objaśnienie oznaczeń:

1. Nagłówek zawierający od prawej numer identyfikacyjny wygenerowanego labiryntu, pod nim przycisk „Show Solutions” służący do wyświetlania rozwiązania, sekcję wyboru algorytmu rozwiązującego „SOLVE ALGORITHM” wraz z przyciskiem wysyłającym żądanie operacji rozwiązania labiryntu, następnie sekcję dodatkowych informacji o wygenerowanym labiryncie zawierającym informacje o szerokości, wysokości, czasie potrzebnym na wygenerowanie labiryntu oraz typie algorytmu użytym do wygenerowania.
2. Plansza labiryntu, która zawiera w górnej części nagłówek ze wskazówką (po najechniu na symbol znaku zapytania) dla użytkownika, informującą o sposobie ustawiania początkowej i końcowej komórki labiryntu, pomiędzy którymi ma być wyszukiwane rozwiązanie oraz pod nagłówkiem wizualizacja labiryntu.

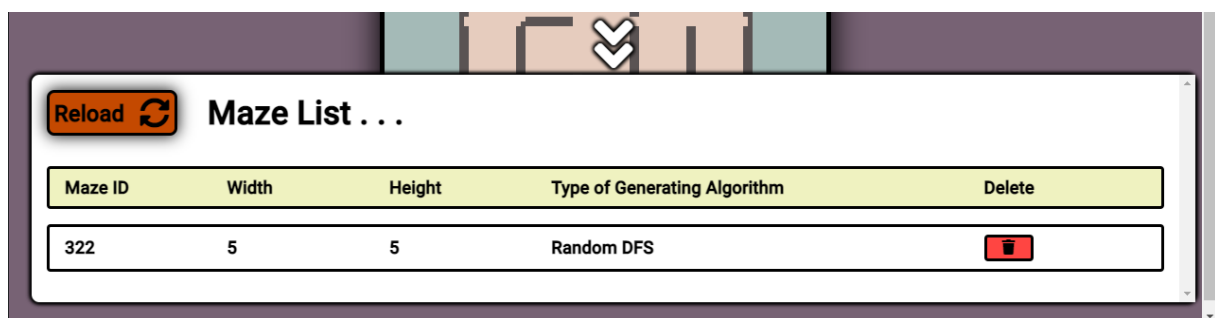
Wskazówka dla użytkownika wraz z przykładowym ustawieniem początkowej i końcowej komórki na siatce labiryntu została zaprezentowana na rysunku 15.



Rysunek 15. Plansza labiryntu ze wskazówką i przykładowym ustawieniem początkowej i końcowej komórki.

Źródło: Autorski rysunek.

Po wygenerowaniu labiryntu do komponentu listy labiryntów został dopisany wygenerowany labirynt. Zaktualizowana lista labiryntów została przedstawiona na rysunku 16.



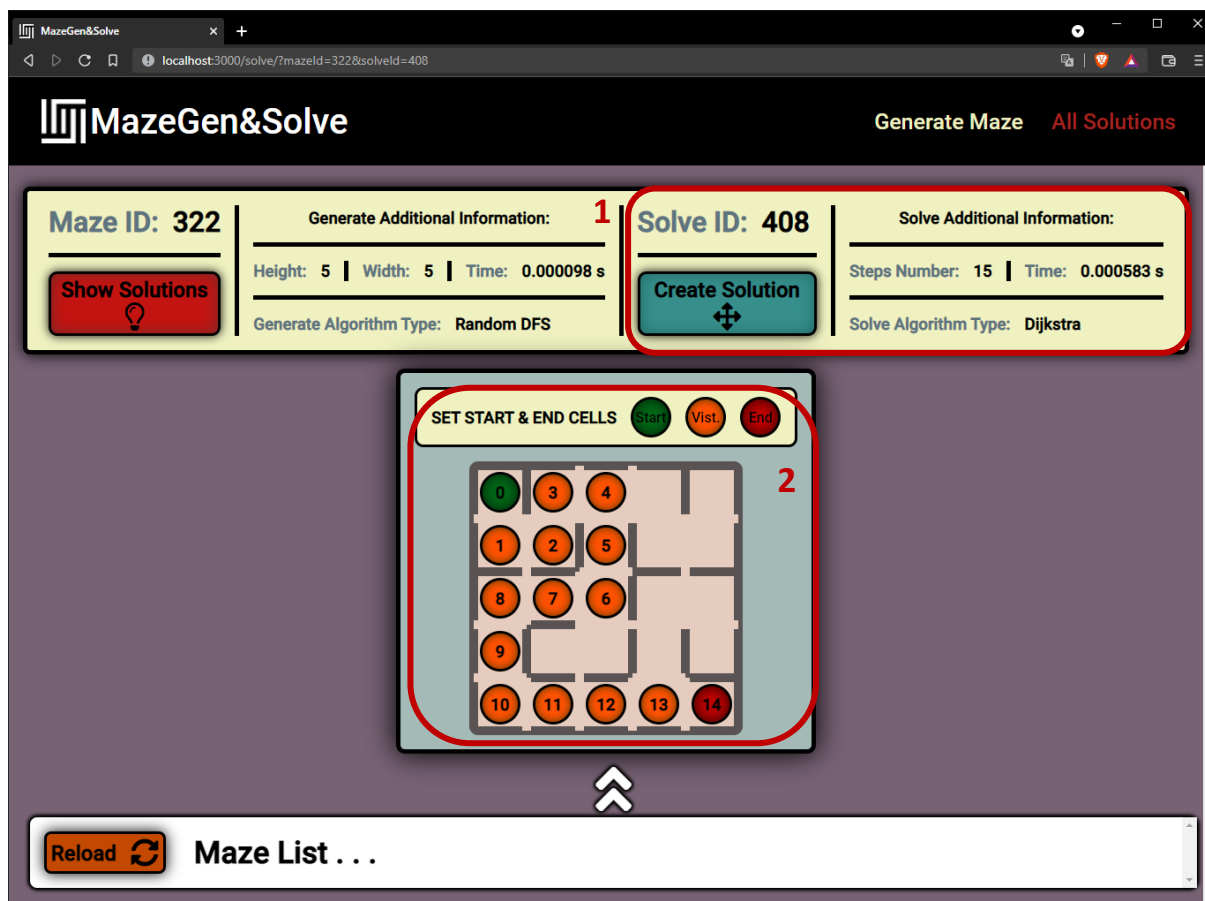
Rysunek 16. Zaktualizowana lista labiryntów. Źródło: Autorski rysunek.

Zwiera ona podstawowe informacje o labiryncie takie jak numer identyfikacyjny labiryntu, szerokość, wysokość oraz użyty algorytm generujący oraz przycisk usunięcia labiryntu z aplikacji z ikoną kosza. Rekordy wyświetlane są w postaci prostokątów, które po naciśnięciu przekierowują na stronę wyświetlającą wygenerowany labirynt.

Po wciśnięciu przycisku „Solve” znajdującego się w nagłówku labiryntu zostaniemy przeniesieni na stronę zawierającą znalezione rozwiązanie zależne od wybranego wariantu.

- Jeśli wybraliśmy komórkę startową i końcową, to rozwiązanie będzie znalezione pomiędzy wybranymi komórkami.
- Jeśli nie wybraliśmy komórki startowej i końcowej, zostaną one wybrane w sposób losowy przez aplikację odpowiednio komórkę startową z komórek znajdujących się w skrajnej lewej kolumnie lub skrajnym górnym wierszu siatki labiryntu, komórkę końcową z komórek znajdujących się w skrajnej prawej kolumnie lub skrajnym dolnym wierszu siatki labiryntu.

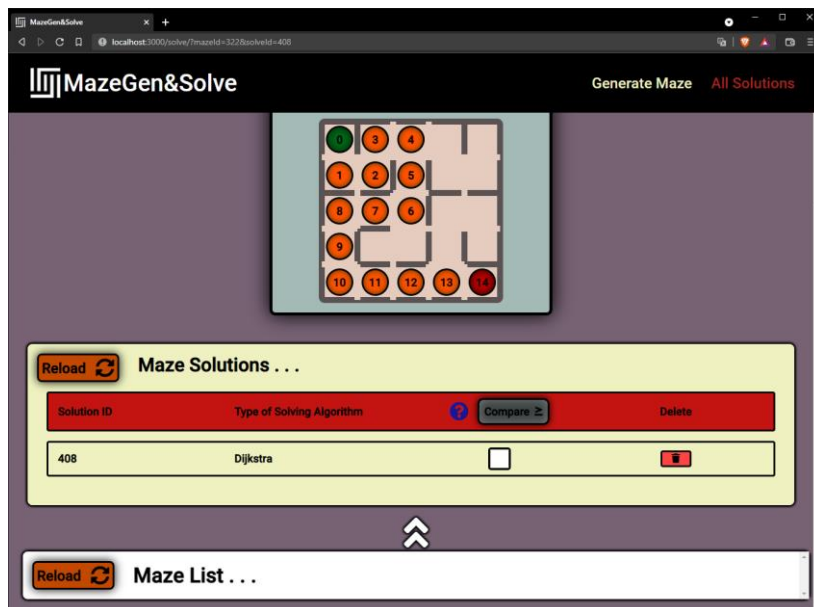
Strona zawierająca rozwiązanie labiryntu została przedstawiona na rysunku 17.



Rysunek 17. Strona rozwiązania labiryntu. Źródło: Autorski rysunek.

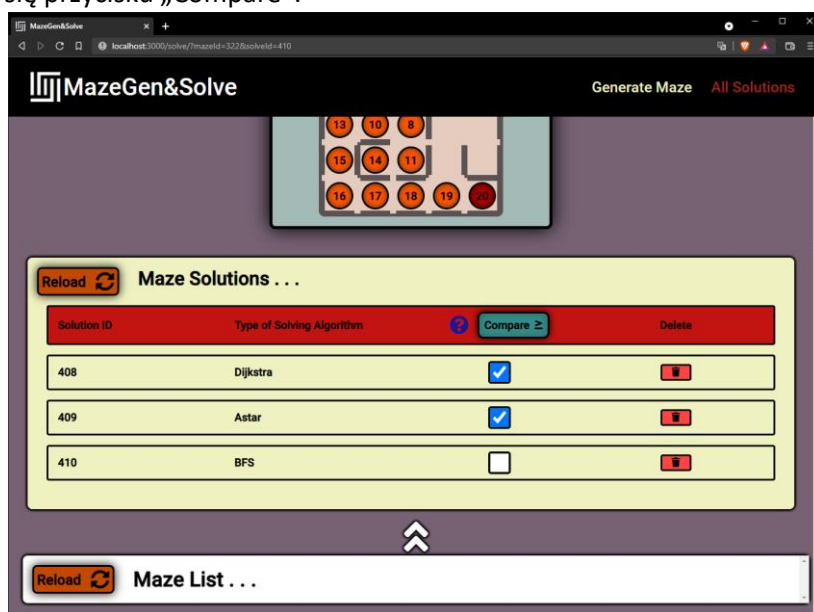
Elementami, które uległy zmianie w stosunku do strony wyświetlającej wygenerowany labirynt jest nagłówek, który zawiera dodatkową sekcję oznaczoną na rysunku numerem 1. Przechowuje ona informację na temat rozwiązania labiryntu takie, jak numer identyfikacyjny labiryntu, przycisk „Create Solution” przekierowujący do strony wyświetlającej wygenerowany labirynt, dla którego jest wyświetlane aktualne rozwiązanie, liczbę kroków potrzebnych do przejścia z komórki początkowej do końcowej, czas potrzebny na wygenerowanie rozwiązania oraz algorytm użyty do rozwiązania. Zmianie uległa również plansza labiryntu, na której znajdują się teraz oznaczenia komórki początkowej (zielona z napisem „Start”), komórek odwiedzonych (pomarańczowa z napisem „Vist.”) oraz końcowej (czerwona z napisem „End”). Na siatce labiryntu komórki zostały oznaczone okręgami o odpowiednim kolorze oraz numerami reprezentującymi kolejne kroki algorytmu, przy czym zerowy numer posiada komórka początkowa a największy komórka końcowa.

Po wygenerowaniu rozwiązania zmienił się stan listy rozwiązań dla labiryntu. Po wciśnięciu przycisku „Show Solutions” w nagłówku labiryntu, pod planszą labiryntu zostanie wyświetlona lista zawierająca rozwiązania związane z rozpatrywanym labiryntem.



Rysunek 18. Lista rozwiązań labiryntu. Źródło: Autorski rysunek.

Lista przekazuje podstawowe informacje o rozwiązaniach takie jak numer identyfikacyjny rozwiązania oraz algorytm użyty do rozwiązania. Z poziomu listy możliwe jest również wybranie opcji usunięcia rozwiązania w postaci ikonki kosza oraz oznaczenie rekordu rozwiązania jako aktualnie wybranego do porównania. Porównanie odbywa się poprzez wciśnięcie przycisku „Compare”. Jest on aktualnie dezaktywowany oraz posiada po swojej lewej stronie ikonkę znaku zapytania, która po najechaniu wyświetla informację o możliwości aktywowania przycisku i opcji porównywania rozwiązań w momencie wybrania minimum dwóch rozwiązań z listy. Liczba możliwych do wybrania rozwiązań na liście nie jest niczym ograniczona. Na rysunku 19 została przedstawiona lista rozwiązań labiryntu zawierająca trzy rozwiązania, z czego dwa pierwsze oznaczone do porównania, co skutkuje uaktualnieniem się przycisku „Compare”.



Rysunek 19. Lista rozwiązań labiryntu z aktywowanym przyciskiem „Compare”. Źródło: Autorski rysunek.

Po naciśnięciu przycisku „Compare” zostajemy przeniesieni na stronę porównania rozwiązań, która została zaprezentowana na rysunku 20.

The screenshot shows the MazeGen&Solve web application interface. At the top, there's a navigation bar with the logo and links for "Generate Maze" and "All Solutions". Below this, the interface is split into two main comparison panels.

Top Panel (Maze ID: 322 vs Solve ID: 408):

- Maze ID: 322**
 - Generate Additional Information: Height: 5 | Width: 5 | Time: 0.000098 s
 - Generate Algorithm Type: Random DFS
- Solve ID: 408**
 - Solve Additional Information: Steps Number: 15 | Time: 0.000583 s
 - Solve Algorithm Type: Dijkstra
- A "Create Solution" button with a crosshair icon is located between the two sections.
- Below the statistics is a "SET START & END CELLS" section with three buttons: "Start" (green), "Visit" (orange), and "End" (red). Below these is a 5x5 grid representing the maze. The grid contains numbered cells (0-14) and some empty cells. The start cell is at (0,0) and the end cell is at (14,4).

Bottom Panel (Maze ID: 322 vs Solve ID: 409):

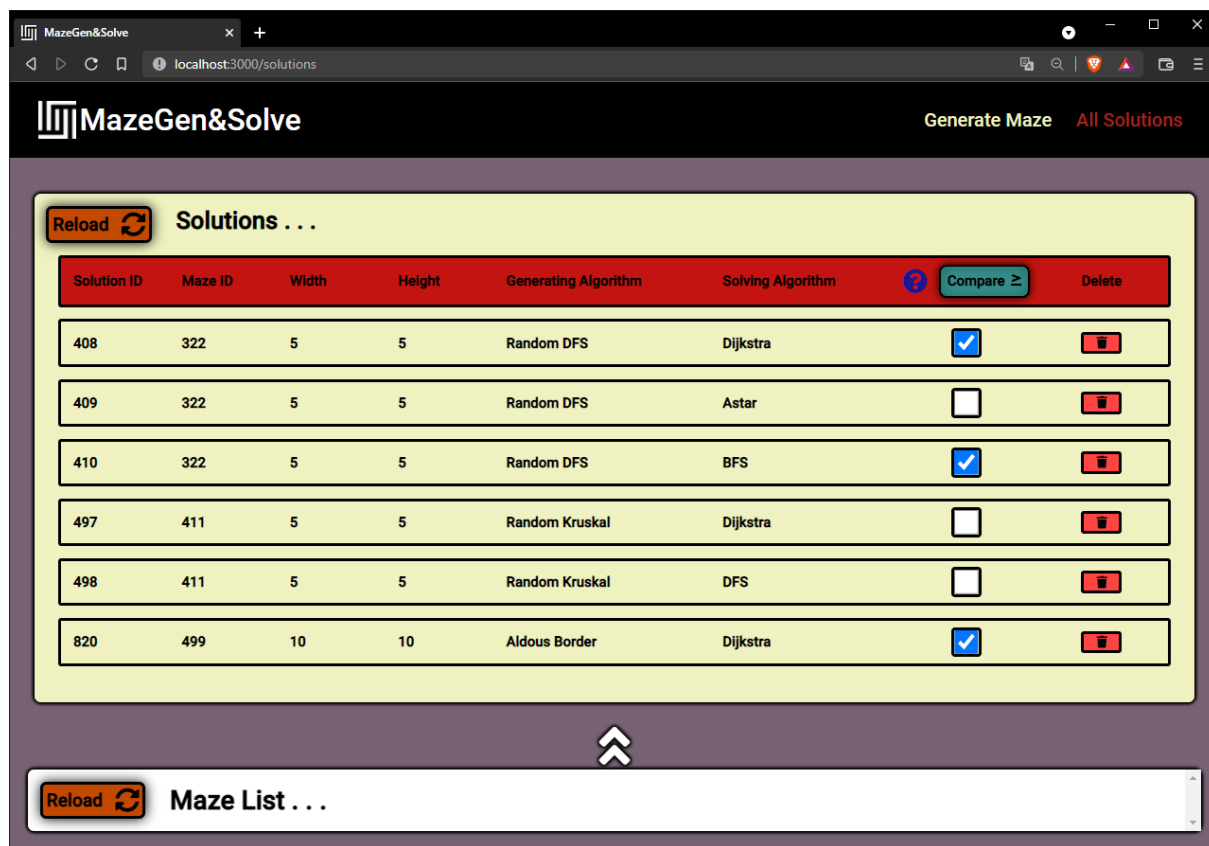
- Maze ID: 322**
 - Generate Additional Information: Height: 5 | Width: 5 | Time: 0.000098 s
 - Generate Algorithm Type: Random DFS
- Solve ID: 409**
 - Solve Additional Information: Steps Number: 21 | Time: 0.002711 s
 - Solve Algorithm Type: Astar
- A "Create Solution" button with a crosshair icon is located between the two sections.
- Below the statistics is a "SET START & END CELLS" section with three buttons: "Start" (green), "Visit" (orange), and "End" (red). Below these is a 5x5 grid representing the maze. The grid contains numbered cells (0-20) and some empty cells. The start cell is at (0,0) and the end cell is at (20,4).

At the bottom of the interface, there is a "Reload" button and a "Maze List ..." link.

Rysunek 20. Przykładowa strona porównania dwóch rozwiązań labiryntu. Źródło: Autorski rysunek.

Możemy na niej zobaczyć zestawienie dwóch rozwiązań w postaci powtórzenia sekwencji: nagłówków oraz plansza labiryntu.

Ostatnią stroną jest strona zawierająca wszystkie rozwiązania labiryntów. Została ona przedstawiona na rysunku 21.



The screenshot shows a web browser window with the URL `localhost:3000/solutions`. The page title is "MazeGen&Solve". There are two main sections: "Solutions ..." and "Maze List ...". The "Solutions ..." section contains a table with the following data:

Solution ID	Maze ID	Width	Height	Generating Algorithm	Solving Algorithm	Compare	Delete
408	322	5	5	Random DFS	Dijkstra	<input checked="" type="checkbox"/>	
409	322	5	5	Random DFS	Astar	<input type="checkbox"/>	
410	322	5	5	Random DFS	BFS	<input checked="" type="checkbox"/>	
497	411	5	5	Random Kruskal	Dijkstra	<input type="checkbox"/>	
498	411	5	5	Random Kruskal	DFS	<input type="checkbox"/>	
820	499	10	10	Aldous Border	Dijkstra	<input checked="" type="checkbox"/>	

Below the table is a "Maze List ..." section with a "Reload" button. The browser's address bar shows `localhost:3000/solutions`.

Rysunek 21. Przykładowa strona wszystkich rozwiązań labiryntów. Źródło: Autorski rysunek.

Zawiera ona rekordy w postaci prostokątów, które przechowują informacje o numerze identyfikacyjnym rozwiązania oraz labiryntu, którego ono dotyczy, szerokości labiryntu, wysokości, algorytmie użytym do wygenerowania oraz rozwiązania labiryntu a także opcję porównywania i usuwania rozwiązań. Po kliknięciu na pole rekordu zostaniemy przekierowani na stronę wyświetlającą wybrane rozwiązanie labiryntu.