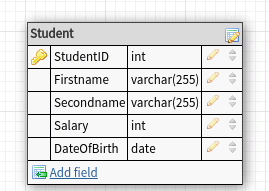
**Zadanie 1**

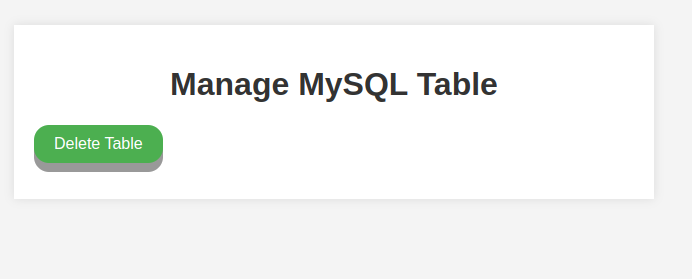
Za pomocą skryptu PHP oraz mysqli utwórz następującą tabelę w bazie danych mysql:



Strona powinna wyświetlić komunikat o utworzeniu tabeli oraz powinna obsłużyć wyjątek, kiedy tabela będzie już utworzona.

Strona powinna posiadać przycisk kasowania tabeli, dzięki któremu usunie się ona z bazy a po jej odświeżeniu utworzy się ponownie.

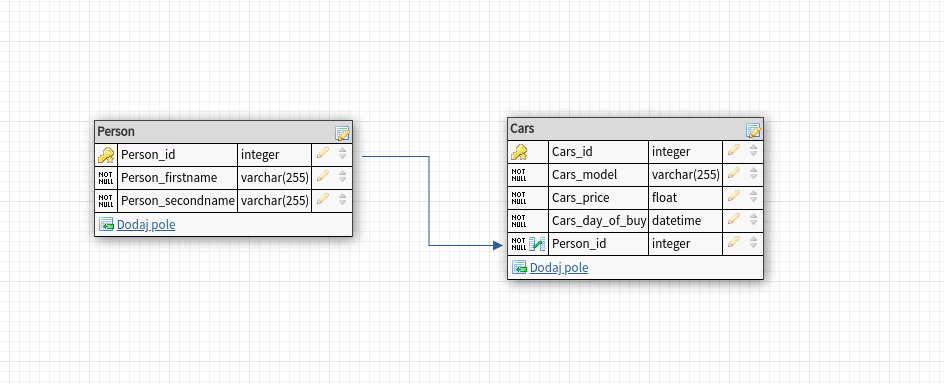
Przykładowy widok strony:



**Zadanie 2**

Na komputerze lokalnym dokonaj podłączenia się z bazą MySQL za pomocą PDO.

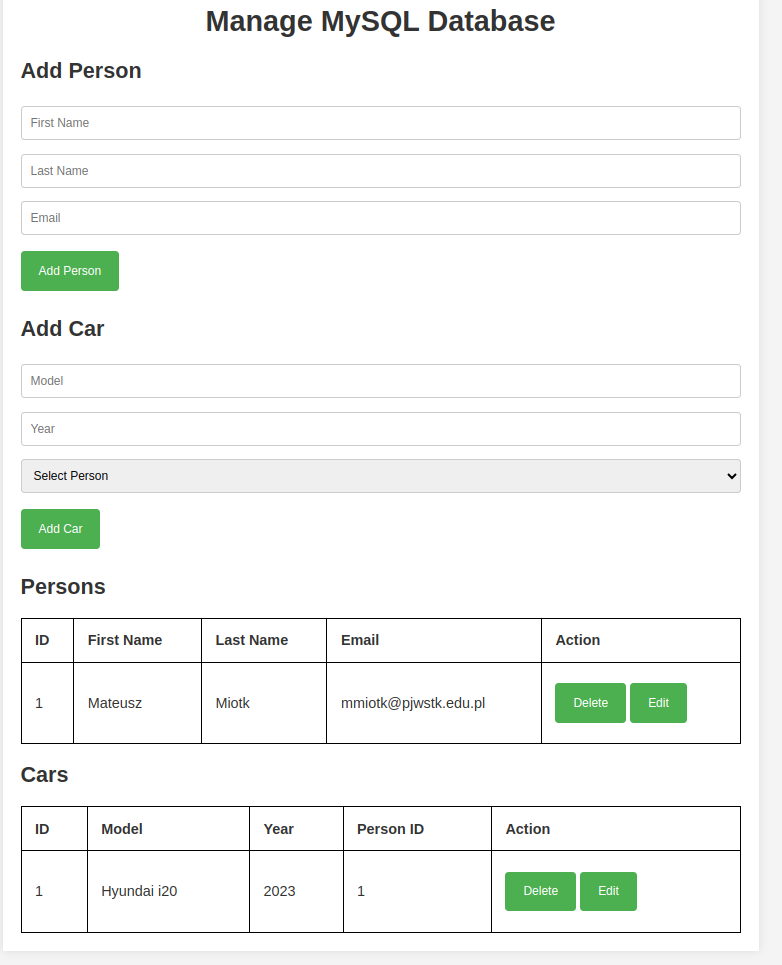
Za pomocą skryptu w PHP utwórz następujące tabele w bazie danych:



Strona powinna wyświetlić komunikat o utworzeniu tabeli oraz powinna obsłużyć wyjątek, kiedy tabela będzie już utworzona. Strona powinna posiadać również następujące funkcjonalności:

* Formularz dodania obiektu klasy Person oraz Cars. W Cars pole Person\_id powinno być wybieralne z listy
* Możliwość wyświetlenia wszystkich danych z obydwu tabel
* Każdy rekord danych powinien posiadać przycisk edycji danych (w wyświetlonych danych). Po naciśnięciu tego przycisku powinien pokazać się formularz z uzupełnionymi danymi, na których chcemy dokonać zmiany
* Każdy rekord danych powinien posiadać przycisk usunięcia danych. Po naciśnięciu przycisku powinien się wyświetlić komunikat potwierdzający.
* Możliwość wyszukiwania wyników tabeli wedle określonego pola

Przykładowy wygląd strony:



**Zadanie 3**

Aby zabezpieczyć hasła użytkowników w bazie danych w języku PHP, możemy do tego użyć funkcji password\_hash:

$password = 'silneHaslo1@';  
$hash = password\_hash($password, PASSWORD\_DEFAULT);  
echo $hash;

Wynik, który jest zwracany z metody password\_hash() jest bezpieczny, ponieważ: wykorzystuje ‎‎silny algorytm szyfrujący (jest to obecnie metodo bcrypt) oraz dodaje ‎‎losową tzw. sól,‎‎ aby zapobiec wyczerpującym atakom i atakom słownikowym.‎ Czyli za każdym razem otrzymamy inny tzw. skrót dla danego hasła. Oto przykłądy wygenerowanych haseł dla powyższego kodu:

$2y$10$G.jBMzXew.PjPE6jNCmmNu6/IUtuUswn00nHn8R7YvVTHjt3ttUfe  
$2y$10$vjCSBzawrEASzLRLYLGCnOqQGWoyFSdgtyqjYxiZocKBCnP3s98Ba  
$2y$10$Z.re.8meKvlXKmzFp1oHcud0JbyXlmjgYcmqTcEtWhLl7DMZrLc/e

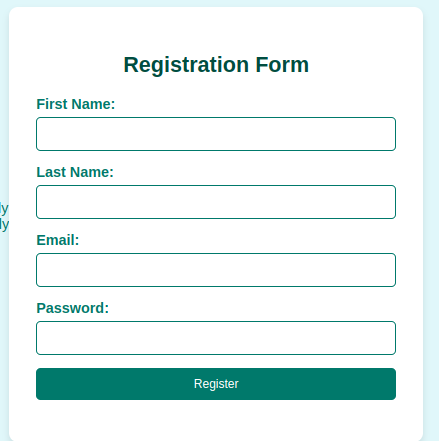
Aby zweryfikować hasło utworzone za pomocą password\_hash() podane przez użytkownika, należy użyć funkcji ‎‎password\_verify().‎ Funkcja ta przyjmuje dwa argumenty:‎ ‎hasło, które powinno być zweryfikowane na poziomie formularza, jako pierwszy argument‎ oraz ‎skrót z funkcji ‎‎password\_hash()‎‎ oryginalnego hasła. ‎Jeśli hasło jest poprawne, ‎‎funkcja password\_verify()‎‎ zwraca wartość ‎‎true‎‎.‎

$password = 'silneHaslo1@'; // Pobieramy te dane z formularza; wcześniej sprawdzamy poprawną formę  
$hash = password\_hash($password, PASSWORD\_DEFAULT); // $hash pobieramy z bazy danych  
var\_dump(password\_verify($password, $hash));

Napisz stronę w PHP, który będzie zawierał formularz rejestracyjny użytkowników do serwisu. Użytkownicy powinni podać co najmniej 5 różnych danych. Strona powinna wyświetlić ilość zarejestrowanych użytkowników i zapisywać ich do utworzonej dla nich bazy danych.

Hasła powinny być kodowane w bazie (za pomocą funkcji haszujących).

Przykładowy wygląd strony:



**Zadanie 4**

Do poprzedniego zadania dopisz formularz logowania się użytkownika do serwisu. Po zalogowaniu użytkownik:

* Może zmienić swoje dane
* Może usunąć konto
* Rozważ sytuację poprawnego logowania oraz niepoprawnego logowania.

Skorzystaj z sesji albo ciasteczek

* Dodatkowo w przypadku utworzenia mechanizmu resetowania hasła dla użytkownika.

Przykładowy wygląd strony:

