**Contacts and Cities Api & Frontend**

There are 2 basic concepts used in this application. These are kept as tables in the database. These concepts;

- Citites

- Contacts

**Purpose of application**

In this application, information on the contacts and cities is kept.

The system shows the contacts and cities as tables. The contact can be added, edited or deleted.

**Sample Scenario:**

1 - The user comes to the contact creation screen.

2 - The user enters the required information and clicks the contact add button. This action will trigger a Add Contact () method in the frontend application.

3 - This action by the user is transmitted to the web api service controller.

4 - The controller looks at the user's request and realizes that it must perform the Create operation.

5 - Subsequently, creating a new contact in the database triggers (frontend / web api / Model relationship).

6 - After the cotactis created, the user will be returned to the contact as listed table on the screen.

ORM (Object Relational Mapper)

The ORM concept is used to easily create the application side Objects in the database.

Contacts

{

public int ID {...};

public string Name {...};

}

When ORM is run for creating on this object, a row with the ID and Name values ​​we have given to the object is created under the Contacts table in the database.

ORM allows us to easily switch between the objects in the application and the database. It acts as a bridge. This makes it very easy to perform CRUD operations on the database.

Business rules

1- User can not enter a contact with empty fields.

**Models**

\* **Contact**

- ID

- Name

- Phone

- Email

-City

\* **City**

- ID

- CityName

**Data Model**

|  |
| --- |
| **Contacts**  ID  Name  Phone  Email  CityID |

|  |
| --- |
| **City**  ID  Name |

Oğuz Kaan Satan

1621221026