In this scenario you have been asked to create a simple REST api and Angular front end to manage a group of users. You will need to create a database and relevant tables to store the information.

The main table should store: A unique ID for the user automatically generated by database, firstname, lastname, Address, Country, Password and Balance. Balance is in EUR

Create a rest api with the following calls:

|  |  |
| --- | --- |
| Register | Accepts username, name, surname, address, country and password. Save data to database and returns the ID, (balance set to 0) |
| Update | Update details of user (except balance). User is identified by the ID sent in api |
| Block | User is identified by the ID sent in api. A blocked user cannot log in. If user is already blocked then a suitable message should be sent back by API. |
| Unblock | User is identified by the ID sent in api. A blocked user cannot log in. If user is blocked then user id unblocked. If user is already unblocked then a suitable message should be sent back by API. |
| Login | Accepts username and password. If user is unblocked and details correct then the userid is returned. |
| Logout | If user is logged in, the user is marked as logged out. |
| AddFunds | Accepts an integer (funds) and adds the amount to the balance. Funds could be negative but balance can never be negative |
| GetBalance | Returns balance for user. User is identified by the ID sent in api. Client can send one of the following currencies as a parameter: USD, EUR , GBP. For USD and GBP the system must send the equivalent of the balance in the currency indicated. The api should use <https://fixer.io/documentation> to display the obtain rates of exchange |

**Consider the following:**

- API Authentication.

- Logging (log all actions with the most possible details; including date, time, user, action)

- Audit (audit all steps taken to undergo an action; including Requests and Responses)

- Unit Tests

- Provide a PostMan Collection which shows the tests conducted on the API.

**Things to keep in mind:**

- [ ] SOLID Principles

- [ ] Validations

- [ ] Error Handling

- [ ] Authentication

- [ ]MsSQL should be used as a database (kindly make sure that all scripts are sent as part of the solution)

**Front End**

Create an angular front end to make use of the API created.

The front end should have the following features:

Features available to unauthenticated users:

1. Register user - collects name, surname, address, country and password / confirm password from user and calls register api
2. Login allows user to enter username and password and calls the Login api

Features available to authenticated users:

1. Logout call logout api
2. Update registered surname, address, country - call update api
3. Update password. Enter old password and new password /confirm new password - call update api
4. Load funds. Allows user to load funds to his account by simply typing amount – call Add Funds
5. Display balance - Display balance in USD, EUR , GBP.