W niniejszym dokumencie opisano działanie oraz niektóre kwestie sporne.

1. Program zbudowany jest z bazy danych oraz trzech modułów:
- credit,
- prodcut,
- customer.

2. Klien komunikuje się z modułem credit, który na porcie 8090 wystawia usługi REST.
2.1 Do stworzenia nowego kredytu usługa przyjmuje wiadomość w formacje JSON na adres: http://localhost:8090/CreateCredit. Przykładowa wiadomoć json wygląda następująco:

```
{
        "customer" : {
                "firstName" : "Jan",
                "lastName" : "Kowalski",
                "personalId" : 1234567890
        },
        "product" : {
     "name" : "motgage credit",
     "value" : 2000
  },
  "credit" : {
                "name": "mortgage"
        }
}
```

W odpowiedzi zwracane jest ID założonego kredytu.

2.2 Drugią dostępną pod adresem http://localhost:8090/GetCredits usługą REST jest przekazywanie wszystkich założonych kredytów. Usługa nie przyjmuje żadnych argumentów, a zwraca listę wszystkich kredytów. Co ważne pojedynczy element listy nie posiada żadnego ID. Wyświetlanie id klientowi jest pierwszą kwestią sporną. ID można dodać dodając adotacje @JsonView(Views.Public.class) w module credit w klasie bazowej credit, product lub customer nad zmienną.

3.Komunikazja z pozostałymi modułami odbywa się na portach:
- 9091 – dla modułu product posiadającego usługi CreateProduct i GetProducts
- 9029 – dla modułu customer posiadającego usłgi CreateCustomer i GetCustomers
- 3306 – dla modułu bazy danych mysql

4. Zarówno usługi CreateProduct, CreateCustomer jak i GetProduct, GetCustomer realizowane są za pomocą protokołu POST. Realizowanie usługi Get za pomocą protokołu POST, nie GET arumentowane jest przesyłaniem dużych ilości numertów ID po dłuższym działaniu aplikacji. Jeżeli zajdzie konieczność można zamienić usługi Get na protokuł GET przesyłając id adresie URI.

5. Użytą bazą danych jest baza mysql na którą można zalogować się poprzez użytkownika:
- użytkownik: ”user” lub “root”
- hasło: “password”

6. Komunikacja z bazą danych została zrealizowana za pomocą JDBC. Jak wiadomo prócz JDBC dostępnej w java.sql Spring posiada własą implementację JDBC. Z powodu niejasności, o którą implementację chodziło zdecydowano się skorzystać z obu. W module product posłużono się implementacją java.sql, a w customer I credit implementacją ze Springa.