Card Cost API

Introduction

The Card Cost API is implemented using the Spring Boot framework and Java 11. Maven is used to manage dependencies and build the application whereas a running MySQL database is used for storage.

Instructions

The following need to be installed:

- Java 11
- Maven
- MySQL Server 8

Steps:

1. Start a MySQL database and create an empty schema for the application (default: cardcostapi)
2. Open the xml configuration file
   source_code/card-cost-api/config/application_properties.xml

   and change the following properties according to your MySQL installation:

   - *spring.datasource.name=cardcostapi*
   - *spring.datasource.url=jdbc:mysql://localhost:3306/cardcostapi*
   - *spring.datasource.username=root*
   - *spring.datasource.password=root*

3. Using a terminal navigate to the ***source_code/card-cost-api*** directory.
4. Run the following command to build the project and run all unit and integration tests (requires database to be running) :

   ```
   mvn clean install
   ```

5. Finally, run the following command to start the Spring Boot application:

   java -jar target/card-cost-api-1.0.0.jar

Security

The Card Cost API only accepts HTTPS connections while it uses a Self-Signed PKCS12 certificate. Furthermore it uses API Key based authentication which dictates that any incoming request must include a header with name **API_KEY** and value **bbc30f15-7e82-48b5-8ac9-e1677c26862d** or an HTTP Forbidden (403) response will be returned.

API Controllers

CountryClearingCostController

Path	/country-clearing-costs/{country_code}
HTTP Method	GET
Description	Returns the clearing cost of the country with code {country_code}.
Response Body	`{ "countryCode": <iso2_code>, "cost": <decimal> }`

Path	/country-clearing-costs
HTTP Method	GET
Description	Returns all existing country clearing costs.
Response Body	`[ { "countryCode": <iso2_code>, "cost": <decimal> }, ... { "countryCode": <iso2_code>, "cost": <decimal> } ]`

Path	/country-clearing-costs
HTTP Method	POST
Description	Creates a country clearing cost using the given request body.
Request Body	`{ "countryCode": <iso2_code>, "cost": <decimal> }`
Response Body	`{ "countryCode": <iso2_code>, "cost": <decimal> }`

Path	/country-clearing-costs/{country_code}
HTTP Method	PUT
Description	Update the clearing cost of the country with code {country_code} using the given request body.

Request Body	`{`
	`"cost": <decimal>`
	`}`

Response Body	`{`
	`"countryCode": <iso2_code>,`
	`"cost": <decimal>`
	`}`

Path	/country-clearing-costs/{country_code}
HTTP Method	DELETE
Description	Delete the clearing cost of the country with code {country_code}.

CardClearingCostController

Path	/payment-cards-cost
HTTP Method	POST
Description	Returns the clearing cost for the card number passed through the request body.
Request Body	`{ "card_number": <pan> }`
Response Body	`{ "country": <iso2_code>, "cost": <decimal> }`

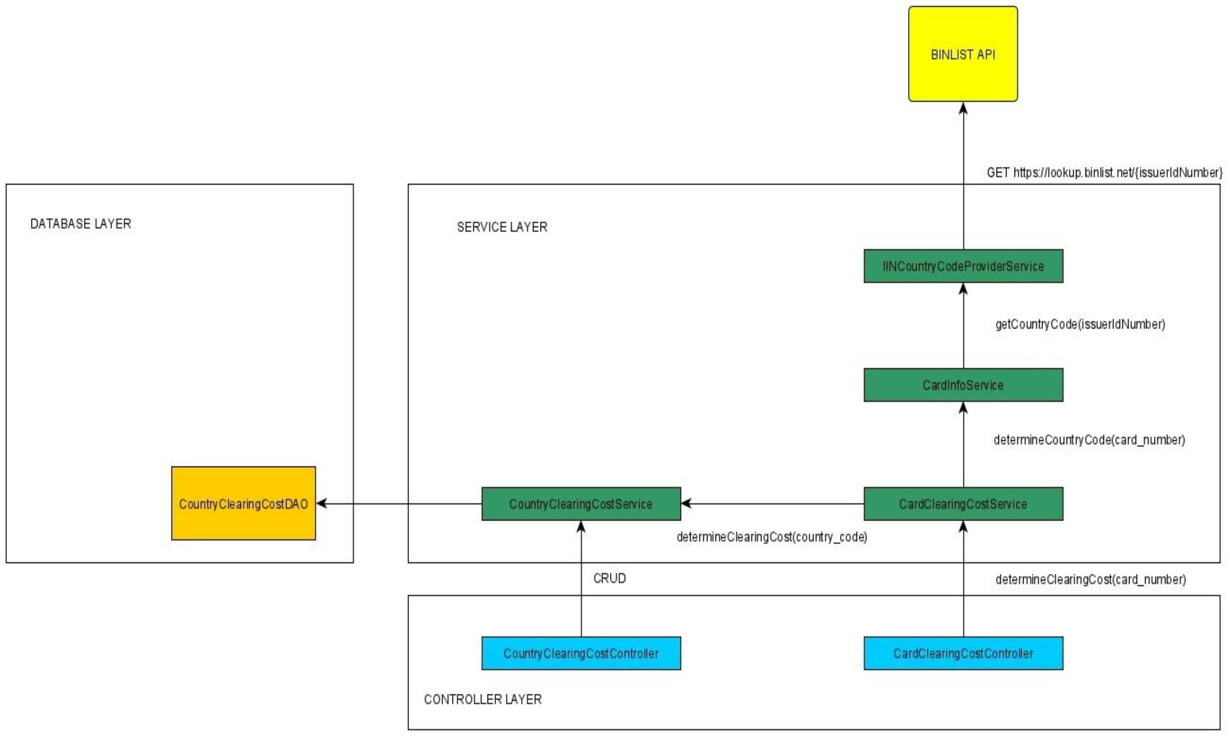
API Errors

Any API Exception is mapped to a response object of the following format:

```
{

    "error_message": <string>,

    "error_code": <string>,

    "timestamp": <local_datetime>

}
```

Error Code	HTTP Status	Description
CCC_01	404	No clearing cost for given country code is found.
CCC_02	400	A clearing cost is already defined for the given country code.
CCC_03	400	Provided country code is invalid.
CCC_04	400	Provided clearing cost is invalid.
CI_01	400	Provided card number is invalid.
BLC_01	404	Country code for the given card number could not be found.
BLC_02	502	Server received an unexpected error while communicating with an external API.
BLC_03	503	Server is under heavy load and cannot handle the request.

API Design Overview


BINLIST API
GET https://lookup.binlist.net/{issuerIdNumber}
DATABASE LAYER
SERVICE LAYER
IINCountryCodeProviderService
getCountryCode(issuerIdNumber)
CardInfoService
determineCountryCode(card_number)
CountryClearingCostDAO
CountryClearingCostService
CardClearingCostService
determineClearingCost(country_code)
CRUD
determineClearingCost(card_number)
CountryClearingCostController
CardClearingCostController
CONTROLLER LAYER

- **CountryClearingCostService**

  Handles the clearing cost matrix CRUD and is able to determine the clearing for a given country code.

- **IINCountryCodeProviderService**

  Provides the country code for a given Issuer Identification Number. Its implementation is the BinlistAPIClient service which communicates with the BINLIST API while also respecting the service SLA (10 requests per minute, with a burst allowance of 10) using a wrapper to Guava's RateLimiter.

- **CardInfoService**

  Validates a card number and utilizes the **IINCountryCodeProviderService** to retrieve the respective country code.

- **CardClearingCostService**

  Uses the **CardInfoService** to retrieve the card number’s country code and the **CountryClearingCostService** to find the clearing cost for the resolved country code.