Fuel consumption application

You are assigned to create fuel consumption management application for small transportation company. It should be possible to report purchased fuel volumes and retrieve the already inserted information by other contributors. The application initially will have only RESTful API with JSON payloads in response, but won't be limited only to this interface in the future. The fuel consumption data should be stored locally in the file or in embedded database, but it should possible later to swap current data storage to something else (MySql, Oracle DB etc).

Fuel consumption registration should accept:

- fuel type(Ex. 95, 98 or D)
- price per litter in EUR (Ex. 10.10
- volume in litters (Ex. 12.5)
- date (Ex. 01.21.2018)
- driver ID (Ex. 12345)

It should be possible to register one single record or bulk consumption (multiple records in one file, for example multipart/form-data).

Data validation should be performed before persisting and corresponding error message should be returned in case of invalid input.

Fuel consumption data retrieval should provide (as a separate endpoint):

- total spent amount of money grouped by month
- list fuel consumption records for specified month (each row should contain: fuel type, volume, date, price, total price, driver ID)
- statistics for each month, list fuel consumption records grouped by fuel type (each row should contain: fuel type, volume, average price, total price)

Every request can be made either for all drivers or for the specific driver (identified by id).

Requirements:

- RESTful API with JSON payloads in response
- Language: Java/Kotlin
- Frameworks: You can use any framework (preferably Spring boot) but keep in mind that usage of framework should be reasonable and architecture of application should allow to switch frameworks.
- Use Gradle or Maven (use dependencies from public repositories)
- Unit tests at least for business logic(use any framework of your choice)
- Use default Maven project structure
- Write production-ready code (clean, modular, testable)
- Apply object oriented design principles