Pluto - simple course management application

User documentation

This is my homework solution for the subject Introduction to Programming 3 at Budapest University of Technology and Economics.

Pluto is a course management Java SE application developed following the MVC design pattern.

This application works like a content management website. In the Pluto system, everything is identified by a Pluto code. After opening the app, you can log in using your credentials (Pluto code and password), or you can register on the registration form page. After that, on the dashboard page you can reach differenc functionalities, all of them are easily understandable and usable. Everything is reachable within clicks and textfield/comboboxes. You can manage your courses taken/your courses instructed, your subjects coordinated, or all the database depending on your user rights.

For more specification see the PLUTO_spec.pdf

Environment

- Developed in JetBrains IntelliJ IDEA 2019.3.3
- Developed with Java SDK 15
  - Uses javax.json 1.4
  - Uses JUnit 5.4
  - Uses dotenv-java 1.2

Packages used:

- (dotenv-java)[https://github.com/cdimascio/dotenv-java] *requires Java 8 or greater*
- (javax.json)[https://mvnrepository.com/artifact/org.glassfish/javax.json/1.1.4]
- (JUnit 5.4)[https://junit.org/junit5/docs/current/user-guide/]

Running

Recommended: Import the project into IntelliJ, set it up for the Application's main class and run it from the IDE. *OR* Import into Eclipse. *OR* Windows:

```
javac @.sources -d out\production\pluto -cp lib\junit-jupiter-5.4.2.jar;lib\junit-jupiter-api-5.4.2.jar;lib\apiguardian-api-1.0.0.jar;lib\opentest4j-1.1.1.jar;lib\junit-platform-commons-1.4.2.jar;lib\junit-jupiter-params-5.4.2.jar;lib\junit-jupiter-engine-5.4.2.jar;lib\junit-platform-engine-1.4.2.jar;lib\javax.json-1.1.4.jar;lib\dotenv-java-2.2.0.jar

java -Dfile.encoding=UTF-8 -classpath out\production\pluto;lib\junit-jupiter-5.4.2.jar;lib\junit-jupiter-api-5.4.2.jar;lib\apiguardian-api-1.0.0.jar;lib\opentest4j-1.1.1.jar;lib\junit-platform-commons-1.4.2.jar;lib\junit-
```

```
jupiter-params-5.4.2.jar;lib\junit-jupiter-engine-5.4.2.jar;lib\junit-platform-engine-
1.4.2.jar;lib\javax.json-1.1.4.jar;lib\dotenv-java-2.2.0.jar pluto.app.Application
```

Unix

```
javac @.sources -d out\production\pluto -cp lib/junit-jupiter-5.4.2.jar;lib/junit-
jupiter-api-5.4.2.jar;lib/apiguardian-api-1.0.0.jar;lib/opentest4j-
1.1.1.jar;lib/junit-platform-commons-1.4.2.jar;lib/junit-jupiter-params-
5.4.2.jar;lib/junit-jupiter-engine-5.4.2.jar;lib/junit-platform-engine-
1.4.2.jar;lib/javax.json-1.1.4.jar;lib/dotenv-java-2.2.0.jar pluto.app.Application

java -Dfile.encoding=UTF-8 -classpath out/production/pluto;lib/junit-jupiter-
5.4.2.jar;lib/junit-jupiter-api-5.4.2.jar;lib/apiguardian-api-
1.0.0.jar;lib/opentest4j-1.1.1.jar;lib/junit-platform-commons-1.4.2.jar;lib/junit-
jupiter-params-5.4.2.jar;lib/junit-jupiter-engine-5.4.2.jar;lib/junit-platform-engine-
1.4.2.jar;lib/javax.json-1.1.4.jar;lib/dotenv-java-2.2.0.jar pluto.app.Application
```

About

- The project is based on the MVC design pattern.
- The structure of the class sulotions is based on the framework of Ruby on Rails (uses MVC as well) and NodeJS, in which I've been working for quite some time now.
- Models communicate with Controllers via exceptions.
- End user communicates/makes commmands through the View to the Controller.
- Controllers get data through a model, that validates and represents the used entities in the db app.
- Controllers are de facto singleton classes, instantiated once, passed only if needed.
- Database is a de jure singleton class, realized its singletonness through the static fields and methods, like the linkedlists it stores during a session.
- Password encryption is secured, but might be prone to memory extractions.
- Environment variables can be set in the `.env` file in the home directory.
    - This way, you can set the admin settings and the debugging settings.
- Project can be found on github
- Developer documentation is in the `/doc` directory and in the source code as Javadoc.
- JSON files:
    - Users > Students have the taken courses attribute for saving them
    - Subject has the coordinator saved
    - Courses have the instructor and parent subject saved
    - This way the database is consistent and has an integrity, from which the loading is clear and unambigous.

Testing

- The project supports unit test of the models.
- After importing the project, you can run the tests with the help of the IDE.