Integration project (30% of the final grade)

Introduction

The essence of the project is to develop a commercial, transactional web application using PHP and a MySQL database. You will develop an online e-commerce shop where users can browse products, add some to a shopping cart and place an order for the cart. We will not implement a real payment system however.
You have free reign on the nature of the online shop, its general design and theme, and the products offered as long as there is more than 5 of them (so at least 6).
This project will be built in steps during weeks 5 to 14 of the semester and completed afterwards.

Timeline

Week # (calendar #)	**General work timeline** *Note: any extra time during a week should be spent on HTML development or PHP business logic development*
1 (5)	Directory structure and general web page layout(s)
2 (6)	Creation of the database structures (tables and such)
3 (7)	Creation of MySQLi query functions in function defining files
4 (8)	None – midterm week
5 (9)	Modelization of objects
6 (10)	Conversion of MySQLi query functions into PDO requests
7 (11)	Implementing PHP sessions
8 (12)	Development of web page HTML and forms
9 (13)	Connection of HTML forms to PHP endpoints through AJAX
10 (14)	Data validation, business logic and completion of AJAX calls
Final exams weeks	Front-end styling, completion of documentation and final tweaks

Requirements

Primary requirements (absolutely necessary for a passing grade):

- The application must built in *PHP*, and designed to be executed on an *Apache* web server.
- A *MySQL/MariaDB* database must be used to store and retrieve data, with *PDO* as the database connection system in *PHP*.
- The application must execute freely of any major execution errors.
- Complete large-scale frameworks, either front-end or back-end (such as *Angular, React, Laravel* or *Symphony*) are forbidden, you must use the standard *PHP* language tools. However, the front-end library *jQuery* is required and the *Bootstrap* one is allowed.
- When submitting the project, the database must be exported as an *SQL* script that allows to set-up a working version on any *MySQL* compliant system, including test data. The script and the entirety of your project directory (and of course, its contents) must be compressed inside a ZIP archive and submitted on LEA
- Obviously, the code must be written by yourself. The use of code-generating tools is forbidden, as well as direct copy of existing code. You are allowed to find references and examples from outside sources, but if you do so, you must re-write it.

Functionality requirements:

- Design and code a single-page application that includes, at the least, the following content pages and functionalities:
  - A login page for client accounts:
    - User registration
    - User login
      - Redirection to the products page on successful login
  - A products listing page
    - List the available products (must be dynamic listing, not based on a fixed number of products), including an image of the product
    - Ability to add a certain quantity of a product to the shopping cart
    - Ability to access the product details page for a product listed
  - A product details page for an individual product
    - Display the full product details, including an image of the product
    - Ability to add a certain quantity of the product to the shopping cart
  - A shopping cart page
    - List the current contents of the shopping cart
    - Modify the quantity of a product in the cart
    - Display the subtotal of the cart
  - An order page
    - Require user login before access
      - Allows login or registration
    - Review the shopping cart/order
    - Enter a billing and shipping address
    - Calculate the order total including taxes and shipping costs
    - “place” an order (without real payment)
  - An order confirmation page
    - Review of a placed order

Modelization

In order to achieve the functionalities above, at least the following classes/objects must be modelized, and a similar representation must be stored in a database:

- Customer
  - id
  - username
  - password hash
  - date created
- Product
  - id
  - display name
  - description
  - image url
  - unit price
  - available quantity
  - date created
- Shopping Cart
  - id
  - status
  - date created
- ShoppingCartProduct
  - product id
  - shopping cart id
  - quantity
- Order
  - id
  - status
  - customer id
  - shopping cart id
  - billing address
  - shipping address
  - date created
  - date placed
  - date shipped

Tertiary requirements

- Use of reusable web page fragments such as header and footers, including a navigation menu
- Project must be linked to a git repository on Github and updates must be pushed regularly
- Git commits must have a clear message representing the committed work
- All file paths used inside your PHP code must be relative and not absolute
- All URLs used inside your HTML web pages must be relative and not absolute
- Use of PDO parameterized prepared statements in database queries
- Use of PHP sessions
- HTML web forms are used to collect user inputs
- Use AJAX asynchronous requests to modify the content of pages
- Use jQuery to implement AJAX requests

- Requests sending data to the server use the POST HTTP method
- HTML pages have viewport-relative size (adapt to user screen size)
- Code is written following certain quality rules:
  - Functions, constants classes and methods should be documented
  - Correct separation of code between public and private directory structures
  - Validation is made on input data (5%)
  - Object security (visibility modifiers) (2.5%)
  - Strong typing of object properties, functions and methods (2.5%)
  - Functions are functionally atomic (5%)
  - Function argument validation (5%)
  - Exception handling (5%)
  - Naming conventions (5%)

Grading

Correction grid:

Graded elements	**%**
Structural requirements *2.1 Proper installation of the Web development platform and the development database management system,* *2.2 Proper installation of software and libraries* • Project built in PHP 8.0+ • Project uses a MySQL / MariaDB • Project runs on an Apache web server • Use of PHP PDO module • Use jQuery to implement AJAX requests	Absolute (failure if not respected)
Respect of required application functionalities *1.1 Accurate analysis of design documents* *1.2 Proper identification of the tasks to be carried out* *3.3 Compliance with the data model* *7.5 Compliance with design documents* • Use of reusable web page fragments • All the required web pages are implemented • All the required web pages implement their required functionalities • All required object types are modelized with realistic attributes • All required models have their database representations (tables)	20
VCS and documentation *2.3 Appropriate configuration of the version control system* *7.4 Compliance with issue tracking and version control procedures* *9.1 Proper identification of the information to be written up* *9.2 Clear record of the work carried out* • Project is linked to a GitHub repository • Git commits must have a clear message representing the commited work • Functions, constants, classes and methods should be documented	10
Database *3.1 Suitable creation or adaptation of the database* *3.2 Proper insertion of initial or test data* *5.3 Appropriate choice of clauses, operators, commands or parameters in database queries* *5.4 Correct handling of database data* *5.5 Appropriate use of data exchange services* • Database script is fully functional • Database script includes initial data • At least 6 products are available in the database • Database queries are parameterized and use prepared statements • Database queries are fully functional • Use of PHP PDO module	20

HTML and front-end *4.1 Appropriate use of markup langage* *4.2 Suitable creation and use of style sheets* *4.3 Proper integration of images* *4.4 Suitable creation of Web forms* *4.5 Adaptation of the interface based on the display format and resolution* *5.2 Proper programming of interactions between the Web interface and the user* *6.5 Web forms in compliance with usability requirements* *6.1 Correct manipulation of DOM objects* *6.2 Proper programming of asynchronous calls* • HTML web forms are used to collect user inputs • HTML web forms are fully functional and collect all the required input data for their tasks • HTML web forms use various and appropriate input elements • User experience (UX) is simple, easy and intuitive • Products have images both in listing and in details • HTML page sections should have viewport-relative sizes • Use AJAX asynchronous requests (calls) with web forms • Use AJAX asynchronous requests to modify the content of pages • AJAX calls sending data to the server use the POST HTTP method • AJAX calls are fully functional • Use of PHP sessions	20
Security and code quality *5.1 Proper programming or integration of authentication and authorization mechanisms* *5.7 Precise application of secure programming techniques* *6.4 Systematic use of Web form data validation techniques* *7.2 Thorough reviews of code and security* *7.3 Relevance of the corrective actions* • Placing an order requires customer account log-in • Customer account passwords are stored as hashed encrypted values • Correct separation of code between public and private directory structures • Validation is made on input data (form-level) • Object security (visibility modifiers) • Strong typing of object properties, functions and methods • Functions are functionally atomic • Naming conventions are respected • Validation is made on input data • Function argument validation • Exception handling	30
TOTAL	**100**