Challenge PHP No.9 - Specification

Description
After the presentation of the challenge and receiving the starter files, each student is required to complete the assignment within the given deadline. The solution should be uploaded to GitLab, with the assessor added as a maintainer to the repository. Additionally, the student is instructed to add the solution link to the learning platform.

Requirements
The goal of this challenge is to create a dynamic website builder. Every visitor that comes to the web application should be able to fill in a couple of predefined fields after which his company's website will be generated. All generated websites will have the same structure, only the basic styling and content that the visitor chooses by themselves are dynamic.

The challenge should have 3 pages and use a MySQL database as data storage (do not use files). You can reference the design from the design folder, feel free to use images of your choice.

When a user clicks on the "Start" button on the home page (see page1.png), he is taken to the second page (see page2.png) where he needs to fill out the following fields:

- Url link for the cover image
- A title for the page
- A subtitle for the page
- A short description of the company
- Telephone number
- Location
- Choose from a dropdown between *services* and *products*
- Three URLs with three descriptions for the three products/services
- Description which will be displayed next to the contact form
- Links for LinkedIn, Facebook, Twitter and Instagram social profiles

Everything entered into this form has to be stored in a **database**. For the structure of the database feel free to design it yourself, just make sure **not to** have columns that will be nullable.
Bonus: Combine OOP with PDO when connecting to the database.

After submitting the form and storing the data, the user is redirected to the third page (see page3.png), where all the information stored in the database, is retrieved and rendered as illustrated on the design for page 3. You pass the information for which record to show via

a GET attribute (example: company.php?id=1, company.php?id=2, company.php?id=3 and so on).

Notes:

- Depending on whether the user selects services or products, that property should be printed in the navbar, the section title, and the title of the card. See page3.png
- Make the navbar items highlighted on hover.
- When the user clicks a particular item in the navbar the page should scroll to the appropriate section.
- The contact form does not have to work (it doesn't need to send emails), just make the design.

Level 1: Beginner

Page 1 & Page 2

Level 2: Intermediate

Page 1 & Page 2 + Page 3

Level 3: Advanced

Page 1 & Page 2 + Page 3 + Database design without nullable columns & using PDO with OOP

Evaluation system

Criteria	Excellent 2.5p	Proficient 2.0p	Good 1.5p	Fair 1.0p	Poor 0.5p
Solution Correctness	*The solution is flawless, meeting all requirements and functionalities.*	*The solution is correct for the most part, with minor errors that do not significantly impact the functionality.*	*The solution is mostly correct, but there are some major errors affecting the overall functionality.*	*The solution has several major errors, making it partially functional, but shows a basic understanding of the problem.*	*The solution is incorrect, incomplete and lacks understanding of the problem.*
Code Quality	*Code is exceptionally well-organized, follows best practices, and demonstrates a deep understanding of all required concepts.*	*Code is mostly clear and well-organized and follows good practices, with only minor improvements needed.*	*Code is mostly organized, but there are notable areas that could be improved for better readability and maintainability.*	*Code lacks organization, readability, and structure, and could be significantly improved.*	*Code is messy, unreadable, poorly organized, and does not adhere to coding standards.*
Adherence to Specifications	*The solution precisely follows all specified challenge requirements.*	*The solution mostly adheres to the specifications, with only minor deviations or oversights.*	*The solution deviates from specifications in some aspects but still fulfills the main requirements.*	*The solution deviates significantly from the specifications, but some elements are implemented correctly.*	*The solution disregards most or all of the specified requirements.*
Documentation, Comments, Cleanliness	*The code is well-documented, clear and exceptionally clean. Comments explain the logic behind any complex parts.*	*Good documentation and comments. Code is generally clean with a few areas for improvement.*	*Basic documentation and comments, with room for improvement in clarity. Code cleanliness is acceptable but needs improvement.*	*Limited documentation and comments. Code lacks clarity and cleanliness and is not well-organized.*	*No documentation or comments. Code is disorganized and challenging to understand.*

Deadline

2 weeks after its presentation, at 23:59 (end of the day).

Assessment Rules

- Fair Assessment: ethical considerations
  - Assessors should ensure that the assessments are conducted in a fair and ethical manner, respecting the principles of academic integrity and honesty.
- Reliability and Validity: enabling consistency
  - Assessments should be consistent and reliable, meaning that they yield consistent results when applied repeatedly to the same task or performance.
  - Assessments must accurately gauge the knowledge, skills, or abilities they are intended to evaluate, ensuring their validity as indicators.
- Feedback: as a method for continuous improvement
  - Assessors should offer constructive feedback that identifies strengths and areas for improvement. The Feedback should be specific and actionable, it should include thought provoking guides and should challenge the student to become better at a specific task.
- Late Submission Policy: assessing assignments after their deadline
  - Students should be allowed a grace period of 3 days (72 hours) to make a late submission on any assignment, with the notice that they will be deducted 20% from the total possible points.
- Plagiarism Policy: assessing assignments with matching solutions
  - In the event of suspected plagiarism, the assessor is required to promptly collaborate with the Student Experience Coordinator/Team as the initial step. Together, they will draft a notice to remind students of the strict prohibition against plagiarism, with potential repercussions for recurrent violations. The following actions will be considered in cases of repeated plagiarism:
    - If a submitted solution exhibits substantial similarity, exceeding 60%, with another student's work (individually or within a group), the respective challenge will incur a 50% reduction from the maximum points attainable.
    - In cases where a solution is identified as more than 90% identical to another student's work (individually or within a group), the challenge in question will receive a score of 0 points.
  - The use of generative AI is encouraged as a learning tool in our educational programs; however, students must engage with the material and contribute with original thought. Reliance on AI for complete content generation is strictly prohibited and will result in point deductions, official warning, or other academic penalties.
  - Upon completion of the assessment process for each challenge/project, the assessor is tasked with selecting the most complete, optimal, and creative solution and to showcase it by publishing it on the platform together with the assessment results
- Timeliness: timeframe for delivering results and feedback to students
  - Feedback on challenges should be provided within 7 days after the deadline has passed.