Компетентності рівня junior для PHP Developer	Загальна оцінка компетентності (max. 18)	Компанії					
		CodeIT	GlobalLogic	Aimprosoft	WEBSPARK	Artjoker	AltexSoft
1. Основи PHP. Базові знання з програмування							
1.1 Змінні та типи даних. Структури даних. Знає основні примітивні типи даних у PHP. Здатний використовувати масиви, включаючи індексовані та асоціативні масиви. Розуміє концепцію NULL і ресурсного типу даних. Вміє ефективно перетворювати типи даних і перевіряти їх, використовуючи вбудовані функції PHP, такі як is_int, is_string та is_array. Вміє створювати та маніпулювати масивами, додавати, видаляти та змінювати їх елементи, а також використовувати функції для роботи з масивами, такі як array_push, array_pop, array_merge, array_filter та array_map. Ідентифікує різницю між примітивними та складними типами даних, а також особливостями роботи з кожним з них. Розуміє області видимості змінних, зокрема, глобальні, локальні та статичні змінні, і знає, як їх використовувати.	**18**	3	3	3	3	3	3
1.2 Оператори та цикли, інструкції мови, умовні конструкції Ідентифікує математичні оператори, логічні, побітові оператори та оператори порівняння. Знає умовні конструкції (if, else, elseif та switch) та вміє застосовувати їх для розробки розгалужень у програмі. Використовує цикли (for, while, do-while та foreach) для ітерації через масиви та інші структури даних, а також повторення блоків коду. Вміє читати та розуміти код, який використовує ці оператори та інструкції. Розуміє, як уникати типових помилок при використанні операторів та циклів, таких як нескінченні цикли та неправильні умови.	**18**	3	3	3	3	3	3
1.3 Функції. Розуміє основні принципи написання функцій, включаючи визначення та виклик функцій. Вміє виділяти частини коду в окремі функції для забезпечення повторного використання та модульності. Володіє навичками ефективного використання глобальних і локальних змінних у функціях, розуміючи, як це впливає на область видимості змінних. Використовує параметри та аргументи функції, параметри по замовчуванню, повернення значень з функції. Застосовує рекурсію для розв'язання завдань, що можуть бути представлені у вигляді повторюваних підзадач. Використовує анонімні функції та колбеки для створення більш гнучкого та динамічного коду Ідентифікує основні патерни проектування функцій, такі як функції вищого порядку, замикання (closures), рефакторинг коду функцій.	**18**	3	3	3	3	3	3
1.4 Рядки. Особливості роботи з кодуванням Розуміє, що таке рядки в PHP і як вони представлені у програмі. Використовує основні операції маніпулювання рядками, такі як конкатенація, розділення та форматування. Знає про різні функції та методи для роботи з рядками, включаючи функції для пошуку та заміни підрядків, обрізання пробілів, отримання довжини рядка та переведення регістру символів. Здатний коректно працювати з різними кодуваннями символів, розуміє, як вони впливають на обробку та відображення рядків у PHP.	**17**	3	2	3	3	3	3
2. Теоретичний блок. Розуміння ООП							
2.1 Основні концепції ООП. Об'єкти. Класи. Властивості та методи Знає принципи використання об'єктів для організації та структурування даних та функціональності у програмі. Ідентифікує розрізнення між класом (шаблоном) та його екземплярами (об'єктами). Ідентифікує властивості (атрибути) і методи (функції) класів, які описують поведінку об'єктів. Розуміє основні концепції ООП (інкапсуляція, спадкування та поліморфізм).	**16**	3	3	2	3	3	2

2.2 Інкапсуляція Усвідомлює, що інкапсуляція дозволяє об'єднати дані та методи класу в єдину одиницю і приховати деталі реалізації. Знає рівні доступу в ООП (public, private, protected) і їхню роль у захисті даних класу. здатний встановлювати та отримувати значення приватних і захищених властивостей через публічні методи (геттери та сеттери). Вміє розробляти класи з урахуванням принципів інкапсуляції для забезпечення безпеки та ефективності коду.	**14**	2	3	2	2	3	2
2.3 Наслідування Розуміє поняття наслідування та його ролі в ООП. Вміє створювати ієрархії класів за допомогою наслідування. Розуміє механізм успадкування властивостей та методів між класами, а також їх перевизначення для адаптації до специфічних потреб програми. Володіє навичками застосування абстрактних класів і інтерфейсів для створення загальних моделей поведінки, які можуть бути реалізовані в різних підкласах.	**14**	2	3	2	2	3	2
2.4 Поліморфізм Ідентифікує поняття та концепцію поліморфізму та його роль в ООП. Вміє створювати універсальні методи та інтерфейси, які можуть обробляти об'єкти різних класів, які реалізують один і той же інтерфейс або наслідують від одного батьківського класу. Вміє використовувати поліморфізм для роботи з об'єктами різних класів через спільний інтерфейс або базовий клас. Розуміє, як поліморфізм використовується у контексті успадкування. Ідентифікує переваги поліморфізму в термінах розширюваності, гнучкості та повторного використання коду.	**14**	2	3	2	2	3	2
2.5 Абстрактні класи та інтерфейси Вміє створювати абстрактні класи і інтерфейси в PHP для встановлення стандартів, які повинні бути реалізовані у підкласах. Здатний визначати загальну структуру класів і методів через абстрактні класи, а також створювати інтерфейси для визначення контракту, який повинні виконувати класи, що імплементують ці інтерфейси. Розуміє, що ці конструкції дозволяють створювати загальні шаблони поведінки для класів, що їх реалізують, і сприяють повторному використанню коду.	**13**	2	2	2	2	3	2
2.6 Простори імен Здатний використовувати простори імен в PHP для організації коду та уникнення конфліктів імен між класами, функціями і константами. Здатний організовувати свій код в логічні групи, використовуючи простори імен для кожної частини програми. Вміє використовувати аліаси для просторів імен для зручного доступу до класів і функцій з інших просторів імен.	**13**	1	2	2	3	3	2
3. Знання Web-технологій. Робота з HTTP-запитами та формами							
3.1 Загальні питання веб технологій Розуміє концепцію клієнт-серверної моделі, ролі клієнта та сервера в архітектурі вебдодатків Знає основні мережеві протоколи, розуміє принцип роботи протоколу HTTP, його основні методи, статусні коди та їх значення. Вміє працювати з заголовками HTTP, формувати HTTP-запити, вказуючи метод, шлях, заголовки та дані. Може обробляти HTTP-відповіді, читати статусні коди, заголовки та вміст відповіді. Розуміє принципи взаємодії із зовнішніми API з використанням HTTP-запитів Має навички використання інструментів розробника браузера для відстеження запитів мережі та аналізу заголовків та вмісту відповідей.	**16**	2	3	3	2	3	3
3.2 Загальна структура HTML-документа. Теги та атрибути. Семантика Знає основні елементи HTML-документу та вміє створювати з них правильну структуру. Знає основні HTML-теги, розуміє їх призначення та особливості, вміє обирати та використовувати найбільш доречні HTML-теги для розмітки різних типів контенту. Використовує семантичні елементи для більш чіткого та структурованого опису вмісту вебсторінки. Розуміє роль та значення атрибутів та їх вплив на поведінку та відображення елементів, вміє додавати та налаштовувати атрибути для елементів HTML. Розуміє важливість валідного HTML-коду та його ролі для сумісності, доступності та оптимізації вебсторінки. Здатний перевіряти та виправляти помилки валідації HTML за допомогою валідаторів та інструментів розробника.	**12**	1	2	3	2	2	2
3.3 Основні CSS-властивості та селектори. CSS-анімації. Flexbox. Grid Розуміє принципи каскадності та спадкування стилів. Вміє працювати з css-властивостями, css-селекторами, та їх ієрархією. Ідентифікує поняття блокової моделі, потоку, позиціонування. Має досвід використання CSS-технік flexbox та grid, вміє налаштовувати гнучку сітку для різних екранів та пристроїв, використовує медіазапити для створення адаптивних вебсторінок. Працює з різними одиницями вимірювання	**6**	1	1	2	1	0	1

3.4 Розуміння роботи API (типи, логіка роботи REST-застосунків) Знає основні принципи REST, такі як ресурси, URL-шляхи, HTTP-методи та параметри запиту. Розуміє структуру URL-шляхів та їх використання для ідентифікації ресурсів і підресурсів. Розуміє рівні доступу та особливості використання ролей користувачів для обмеження доступу до ресурсів.	**13**	2	3	2	2	2	2
3.5 Робота з HTTP-запитами. Обробка запитів GET та POST, робота з параметрами запиту. Вміє обробляти GET і POST-запити в PHP, отримуючи доступ до параметрів запиту за допомогою глобальних масивів $_GET і $_POST. Здатний отримувати значення параметрів, перевіряти їх наявність і виконувати необхідні дії в залежності від цих значень. Усвідомлює, що GET-параметри можуть бути відкрито доступні у URL і обмежені за розміром, тоді як POST-параметри передаються через тіло запиту і можуть містити більші обсяги даних.	**17**	2	3	3	3	3	3
3.6 Робота з формами. Обробка даних, введених користувачем через HTML-форми в PHP-сценаріях Розуміє структуру HTML-форм, включаючи елементи <form>, <input>, <textarea>, <select> та інші, які використовуються для збору текстових, числових і вибіркових даних. Здатний правильно валідувати та очищати дані, введені користувачем, перед їхнім використанням у власній програмі. Вміє обробляти дані, введені користувачем через HTML-форми в PHP-сценаріях, отримуючи доступ до введених даних за допомогою глобального масиву $_POST (для методу POST) або $_GET (для методу GET).	**16**	2	3	2	3	3	3
3.7 Сесії та куки: використання та керування Розуміє основні концепції сесійних механізмів та куків, їх взаємодію з HTTP-протоколом і роль в збереженні стану між візитами користувачів. Ідентифікує, що сесії зберігають дані на сервері, а куки — на боці клієнта, і можуть бути використані для збереження ідентифікаційної інформації, налаштувань користувача або іншої інформації, яка повинна зберігатися між різними візитами на веб-сайт. Здатний створювати, зберегати та видаляти сесійні дані для зберігання тимчасових даних індивідуального користувача. Вміє створювати, зчитувати та видаляти куки на стороні клієнта, налаштовувати термін дії куки і область видимості.	**14**	1	3	3	2	3	2
3.8 Робота з Ajax. Генерація сторінки в форматі JSON. Розуміння та обробка даних у форматах XML і JSON. Знає основні концепції та принципи роботи з Ajax, такі як асинхронні запити, обробка відповідей на стороні клієнта, використання XMLHttpRequest або Fetch API. Розуміє, як використовувати PHP для генерації динамічного контенту у форматі JSON для асинхронних запитів. Розуміє відмінності між форматами XML і JSON, а також їхні переваги та недоліки в різних контекстах. Знає, як читати та створювати дані в обох форматах.	**11**	1	1	2	2	3	2
4. Робота за базами даних							
4.1 Робота з базами даних MySQL.Підключення до бази даних, PDO. Розуміє основні принципи реляційних баз даних, такі як структура таблиць, індекси, зв'язки між таблицями. Використовує бази даних MySQL для зберігання і організації даних у веб-додатках, здатний створювати та модифікувати таблиці, додавати індекси та встановлювати зв'язки між таблицями для забезпечення цілісності даних. Вміє підключатися до бази даних MySQL з використанням універсального інтерфейсу PDO, встановлювати необхідні атрибути підключення та обробляти помилки підключення.	**17**	3	3	3	2	3	3
4.2 Знання мови SQL Здатний виконувати основні SQL-запити для роботи з базами даних, такі як вибірка (SELECT), вставка (INSERT), оновлення (UPDATE) і видалення (DELETE) даних. Може створювати складні запити з використанням умов (WHERE), сортування (ORDER BY), групування (GROUP BY), а також об'єднання таблиць (JOIN) для отримання необхідних даних з різних таблиць. Володіє навичками оптимізації запитів за допомогою індексів та розуміє, як уникати надмірного навантаження на базу даних.	**14**	3	2	2	2	3	2
4.3 Адміністрування баз даних через веб-інтерфейс PhpMyAdmin Використовує PhpMyAdmin для адміністрування баз даних MySQL. Вміє імпортувати та експортувати дані, виконувати SQL-запити і налаштовувати права доступу користувачів. Розуміє, як використовувати PhpMyAdmin для швидкого виконання адміністративних завдань без необхідності написання SQL-коду вручну. Володіє навичками створення резервних копій баз даних і відновлення даних з резервних копій для забезпечення безперервної роботи системи.	**9**	2	2	1	1	2	1

4.4 Транзакції та управління ними Усвідомлює важливість транзакцій у забезпеченні цілісності і консистентності даних у базі даних, а також їх властивості ACID (Atomicity, Consistency, Isolation, Durability), які гарантують надійність виконання операцій. Вміє починати, підтверджувати (commit) та відміняти (rollback) транзакції в MySQL за допомогою PHP PDO. Здатний обробляти транзакції у PHP-коді, забезпечуючи правильне виконання всіх необхідних операцій або відкат у випадку помилки. Знає різні рівні ізоляції транзакцій і їх вплив на паралельність і консистентність даних (Read Uncommitted, Read Committed, Repeatable Read та Serializable), та як вони можуть допомогти уникнути проблеми "брудного читання", "неповторюваного читання" і "фантомного читання".	**11**	2	2	1	2	2	2
5. Розширені знання PHP							
5.1 Робота з файлами. Операції з файлами та каталогами на сервері. Знає основні функції PHP для роботи з файлами та каталогами. Розуміє різницю між режимами відкриття файлів (r, w, a і т.д.) та їх використання. Вміє створювати, перейменовувати та видаляти файли і каталоги на сервері. Розуміє важливість прав доступу до файлів та каталогів і знає, як їх змінювати за допомогою функції chmod(). Використовує функції PHP для роботи з великими файлами, обробляючи їх частинами для уникнення перевантаження пам'яті. Знає функції для завантаження файлів з клієнтської частини та вміє обробляти ці завантаження на сервері, забезпечуючи перевірку типу файлу та захист від шкідливих файлів.	**12**	1	2	2	2	3	2
5.2 Потоки вводу-виводу: Робота з потоками вводу-виводу для обробки даних в реальному часі. Знає основні концепції потоків вводу-виводу, такі як стандартний вхід (stdin), стандартний вихід (stdout) та стандартний помилковий вихід (stderr). Розуміє важливість потоків для обробки даних у реальному часі, зокрема для читання та запису даних з файлів, мережевих з'єднань та інших ресурсів. Вміє використовувати функції PHP для керування потоками вводу-виводу, такі як встановлення блокуючого чи неблокуючого режиму, перевірка стану потоків, моніторингу кількох потоків одночасно та реагування на їх події, а також використання їх для обробки даних у реальному часі.	**8**	1	1	1	1	2	2
5.3 Функції для роботи з датами та часом: Використання функцій для роботи з датами та часом, форматування дат, обчислення різниці між датами. Знає основні функції для роботи з датами та часом. Розуміє різницю між різними форматами дат і часів та їх використання у веб-додатках. Вміє форматувати дати у різні формати відповідно до вимог, обчислювати різницю між датами, працювати з часовими зонами, встановлювати та змінювати їх. Розуміє принципи роботи з датами у форматі Unix timestamp і здатний конвертувати їх у різні формати Використовує клас DateTime та його методи для роботи з більш складними операціями над датами та часом, такими як додавання або віднімання інтервалів	**12**	2	1	2	2	3	2
5.4 Функції для роботи з регулярними виразами Знає основні функції та синтаксис для роботи з регулярними виразами в PHP, використовує регулярні вирази для пошуку та обробки тексту, а також у валідаторах форм. Розуміє основні концепції регулярних виразів, такі як метасимволи, квантифікатори, спеціальні послідовності, групи та інші конструкції.	**11**	2	1	1	2	3	2
5.5 Обробка помилок та винятків. Знає вбудовані функції та механізми PHP для обробки помилок, такі як try, catch, використовує throw для роботи з винятками (exceptions) та функції для роботи з помилками. Розуміє різницю між винятками (exceptions) та звичайними помилками (errors), а також використання кожного механізму відповідно до ситуації. Вміє налаштовувати рівень звітності про помилки. Вміє створювати власні класи винятків для конкретизації типів помилок, а також власні обробники помилок для спеціалізованої обробки помилок у власних PHP програмах	**14**	3	3	1	2	3	2
5.6 Основи безпеки (захист від SQL-ін'єкцій, XSS-атак, CSRF-захист, хешування паролів) Знає загрози безпеки, такі як SQL-ін'єкції і XSS-атаки, і про базові методи захисту від них. Вміє використовувати підготовлені SQL-запити з PDO для запобігання SQL-ін'єкціям. Вміє фільтрувати та екранувати дані перед виведенням у HTML для запобігання XSS-атак. Використовує токени CSRF для перевірки автентичності запитів, що надходять на сервер, щоб уникнути виконання небажаних операцій без дозволу користувача. Використання безпечні методи зберігання паролів (наприклад, з хешуванням та солюванням), а також застосовує SSL/TLS для шифрування передачі даних між клієнтом і сервером, що забезпечує конфіденційність та цілісність інформації під час передачі через мережу.	**14**	2	3	2	2	3	2

5.7 Основи тестування за допомогою PHPUnit Знає основи юніт-тестування, вміє інсталювати та налаштовувати PHPUnit, розуміє структуру тестових класів та методів. Вміє писати тести для перевірки окремих функцій і методів у коді, використовує асерти для перевірки очікуваних результатів. Розуміє принципи тестового використання, включаючи структуру Arrange-Act-Assert (AAA), роботу з моками (mocks) і стабами (stubs) для імітації залежностей і зовнішніх сервісів під час тестування.	**9**	1	2	1	2	1	2
5.8 Робота з популярними CMS (WordPress, Magento, OpenCart) Знає основні концепції і можливості вибраної CMS, такі як управління контентом, робота з шаблонами і плагінами, адміністрування користувачів і прав доступу. Вміє встановлювати і налаштовувати вибрану CMS, створювати та редагувати веб-сайти, використовуючи її функціонал для реалізації базових потреб клієнта чи проєкту Розуміє основні принципи безпеки і оптимізації вибраної CMS, а також можливості розширення її функціоналу за допомогою плагінів чи модулів.	**5**	0	1	1	0	2	1
5.9 Розуміння одного з фреймворків PHP (Laravel, Symfony) Знає основні поняття і концепції вибраного фреймворка, такі як структура проєкту, конфігурація, робота з пакетами і залежностями. Вміє встановлювати і налаштовувати фреймворк, розробляти інтерфейси користувача через роутинг, контролери і представлення (шаблони), виконувати базові CRUD операції з даними. Розуміє механізми інтеграції з базами даних, безпеки, автентифікації та авторизації, а також можливості розширення функціоналу за допомогою розширень та плагінів.	**11**	1	3	1	2	2	2
6. Інструментарій розробника							
6.1 Системи контролю версій - GIT Розуміє git flow - методологію роботи з гілками для ефективного управління процесом розробки. Знає основні концепції версіонування коду, такі як коміти, гілки, злиття. Здатний створювати репозиторії та ініціалізувати проєкти з використанням Git. Розуміє та використовує команди Git (commit, push, pull, fetch, checkout, add, status, revert та інші). Вміє працювати з гілками (branches) для розробки функціональності відокремлено від основної гілки (наприклад, розробка нових функцій у власних гілках та їх об'єднання в основну гілку). Знайомий з процедурою створення пулл-реквестів для обговорення та злиття змін з різних гілок.	**15**	2	3	2	2	3	3
6.2 Навички використання IDE Вміє працювати з популярними IDEдля розробки PHP, такі як PhpStorm, Visual Studio Code, NetBeans. Використовує редактор коду, налаштування розширень та плагінів для поліпшення продуктивності розробки. Застосовує вбудовані інструменти IDE для налагодження коду, аналізу помилок, рефакторингу та контролю якості коду.	**9**	2	2	2	1	0	2
6.3 Навички використання командного рядка (Terminal) Має базове розуміння навігації по файловій системі через командний рядок. Використовує команди для перегляду, створення, зміни, видалення та архівації файлів та тек через командний рядок. Застосовує через командний рядок команди для роботи з Git, таких як клонування репозиторію, створення гілок, комітів та інші.	**11**	2	2	1	1	3	2
6.4 Знання Linux-середовища, Docker Розуміє основні концепції та принципи роботи з операційною системою Linux. Вміє працювати з командним рядком Linux, виконувати команди, керувати файловою системою, управляти користувачами та правами доступу. Працює зі скриптовими мовами, такими як Bash, для автоматизації рутинних завдань та конфігурації середовища. Вміє створювати, налаштовувати та управляти Docker контейнерами для окремих мікросервісів. Знає Docker-команди для роботи з образами, контейнерами, мережами, томами та іншими компонентами Docker.	**9**	2	2	1	1	2	1
6.5 Встановлення PHP та налаштування серверу (XAMPP, WAMP, MAMP). Принципи роботи хостінгу. Знає процес встановлення PHP, Apache (або Nginx), MySQL (або іншої СУБД), конфігурування середовища розробки. Вміє налаштовувати PHP, встановлювати та конфігурувати XAMPP, WAMP, MAMP для локальної розробки і тестування. Розуміє принципи роботи хостингу, підключення доменів та налаштування хостингу. Здатний здійснювати підключення по FTP, FTPS. Вміє налаштовувати php.ini Вміє конфігурувати і налаштовувати параметри сервера, такі як порти, шляхи до файлів і модулі PHP.	**7**	2	1	2	1	0	1
7. Розмовна англійська							

7.1. Правильна побудова фрази з узгодженням часу Вміє використовувати відповідні часові форми дієслів і структури речень для точного вираження часу в комунікації. Здатний правильно використовувати часові форми, такі як Present Simple, Present Continuous, Past Simple, Past Continuous, Future Simple, для передачі правильного значення в повідомленні.	**14**	2	3	3	2	2	2
7.2 Професійна it термінологія в розмові з замовником Знає і використовує професійні IT-терміни та скорочення, які використовуються в IT-індустрії, для чіткого та зрозумілого спілкування зі замовником. Вміє перекладати складні технічні концепції на доступну мову для замовника, не знайомого з IT-галуззю.	**10**	2	2	2	0	2	2
7.3 Професійна it термінологія у діловому звіті Знає і використовує спеціалізовану IT-термінологію та фразеологію при написанні ділових звітів. Вміє передати інформацію про технічні питання та проєкти зрозуміло та конкретно, використовуючи адекватні IT-терміни та абревіатури.	**10**	2	2	1	0	2	3
7.4 Професійна термінологія для PHP Знає та розуміє специфічну термінологію, що використовується в контексті програмування PHP Розуміє та використовує специфічні терміни, патерни, алгоритми, фреймворки, бібліотеки та інші інструменти, які використовуються в PHP	**13**	3	3	2	1	2	2
7.5 Неформальне спілкування (small talks) Вміє вести неформальну розмову, розпочинати діалоги та підтримувати невимушену атмосферу. Знає загальновживану термінологію та фрази, які використовуються у неформальних розмовах, таких як загальні питання про погоду, цікаві події, особисті захоплення, спорт, фільми тощо.	**7**	2	1	1	0	2	1

Коментарі до таблиці

Оцінки компетентностей вказані станом на 2024/2025 рр., по шкалі від 0 до 3.

«3» - компетентність вкрай важлива, якщо кандидат нею не володіє, його подальше просування на вакансію

«2» - компетентність, знання якої обов'язково знадобиться кандидату при подальшому професійному зростанні.

«1» - некритична компетентність, якою можна знехтувати.

«0» - неактуальна компетентність.

Зауваження від компаній

CodeIT

2.6 Як для джуніор розробника така навичка буде плюсом, але не є критичною. Гарно побудований процес розробки у команді та менторинг дозволить доволі швидко розвити цю компетенцію.

GlobalLogic

1. Основи PHP. Базові знання з програмування

1.1. Володіння основними типами даних, масивами та розуміння областей видимості змінних є обов'язковими для Junior PHP Developer.
1.2. Оператори, цикли та умовні конструкції є базовими знаннями та необхідними для розуміння основ програмування.
1.3. Розуміння функцій, параметрів, рекурсії, замикань, а також роботи з локальними та глобальними змінними є важливою компетенцією.
1.4. Робота з рядками є важливою навичкою, але не є критичною вимогою для Junior-рівня.

2. Розуміння ООП

2.1. Розуміння класів, об'єктів, властивостей, методів та основних принципів ООП (інкапсуляція, наслідування, поліморфізм) є критично важливим для Junior PHP Developer.
2.2. Робота з інкапсуляцією методів та властивостей є необхідною для створення структурованого коду.
2.3. Наслідування, абстрактні класи та інтерфейси є важливими для розуміння побудови програмних рішень.
2.4. Поліморфізм є важливою концепцією, яку бажано розуміти на рівні Junior.
2.5. Абстрактні класи та інтерфейси є корисними, але їх глибоке розуміння не є критичною вимогою для початкового рівня.
2.6. Простори імен важливі для організації коду, але достатньо базового розуміння.

3. Знання Web-технологій. Робота з HTTP-запитами та формами

3.1. Розуміння вебтехнологій та принципів роботи HTTP є необхідним для взаємодії з вебсервісами та API.
3.2. Базові знання HTML є корисними для веброзробки, але не є ключовою вимогою для PHP Developer.
3.3. Основи CSS є перевагою, але не є критичною компетенцією для серверної розробки.
3.4. Інтеграція з API є важливою компетенцією для сучасного PHP Developer.
3.5. Розуміння GET, POST-запитів та передачі параметрів є базовою необхідною навичкою.
3.6. Робота з HTML-формами, валідація та очищення даних є важливими для створення безпечних застосунків.
3.7. Розуміння роботи із сесіями та cookies є важливою компетенцією для Junior PHP Developer.
3.8. Базове розуміння AJAX та роботи з JSON є бажаним, але не є критичною вимогою.

4. Робота з базами даних

4.1. Робота з MySQL та PDO є критично важливою компетенцією для Junior PHP Developer.
4.2. Достатньо знати основи SQL: вибірку, вставку, оновлення та видалення даних. Глибока оптимізація запитів та робота з індексами не є обов'язковими.
4.3. Робота з PhpMyAdmin не є критичною вимогою, оскільки адміністрування баз даних часто належить до інших напрямів.
4.4. Базове розуміння транзакцій є необхідним, але детальне знання рівнів ізоляції та внутрішніх механізмів не є обов'язковим.

5. Розширені знання PHP

5.1. Робота з файлами є корисною навичкою, але не є критичною для базових завдань.
5.2. Базове розуміння потоків вводу-виводу є перевагою, але не є обов'язковим.
5.3. Робота з датами та часом є важливою практичною навичкою, але не є критичною вимогою.
5.4. Знання регулярних виразів є перевагою, але не є обов'язковим для основних задач.
5.5. Обробка помилок та винятків є важливою компетенцією для створення надійного коду.
5.6. Основи безпеки вебзастосунків є критично важливими для Junior PHP Developer.
5.7. Знання PHPUnit є перевагою, але не є обов'язковим для початкового рівня.
5.8. Робота з CMS (WordPress, Magento, OpenCart) залежить від специфіки проєкту.
5.9. Розуміння одного з PHP-фреймворків (Laravel, Symfony) є важливим для комерційної розробки.

6. Інструментарій розробника
6.1. Знання Git є критично важливою компетенцією для Junior PHP Developer.
6.2. Навички роботи з IDE є важливими, але не є критичними для базових завдань.
6.3. Базові навички роботи з Terminal є бажаними, але не обов'язковими.
6.4. Знання Linux та Docker є перевагою, але не є критичною вимогою.
6.5. Вміння налаштувати локальне середовище розробки є корисним, але не є обов'язковим рівнем глибокої експертизи.

7. Розмовна англійська
7.1. Розуміння граматики та правильна побудова речень є важливими для ефективної комунікації.
7.2. Технічна термінологія є перевагою, але не є критичною вимогою.
7.3. Навички написання технічної документації та звітів є бажаними.
7.4. Знання PHP-термінології є важливим для роботи з документацією та командою.
7.5. Неформальне спілкування англійською не є критичною компетенцією для Junior PHP Developer.

Aimprosoft

1. Основи PHP. Базові знання з програмування
1.1. Змінні, типи даних та структури даних є базовою та обов'язковою компетенцією для Junior PHP Developer. Функції для роботи зі строками, масивами та іншими структурами важливо не заучувати, а розуміти принципи використання та вміти знаходити необхідну інформацію в документації.
1.2. Оператори, цикли, інструкції мови та умовні конструкції є обов'язковими базовими знаннями. У відповідях бажано наводити приклади використання різних типів циклів та пояснювати, у яких ситуаціях кожен з них доцільний.
1.3. Функції є важливою частиною базових знань. Для Junior-рівня допускається часткове нерозуміння або відсутність досвіду з closures, anonymous functions та callbacks за умови теоретичного розуміння їх призначення.
1.4. Робота зі строками та особливості кодування є обов'язковою темою. Це також є природним переходом до регулярних виразів, пошуку підрядків та роботи з масками.

2. Розуміння ООП
2.1. Основні концепції ООП (класи, об'єкти, властивості та методи) є обов'язковими. Для Junior-рівня допускаються неточності через недостатній практичний досвід, однак теоретичне розуміння має бути сформованим.
2.2. Інкапсуляція оцінюється не лише через визначення, а через розуміння її призначення. Важливо, щоб кандидат міг пояснити, навіщо вона потрібна, і навести практичні приклади використання.
2.3. Наслідування є ключовою концепцією ООП. Бажано також розуміння множинного наслідування та способів його реалізації (наприклад, через interfaces або traits). Важливі практичні приклади.
2.4. Поліморфізм є однією з найскладніших тем для Junior-рівня. Основний фокус — розуміння мети його використання та практичного застосування.
2.6. Простори імен (namespaces) мають бути зрозумілі на теоретичному рівні: визначення та призначення є достатніми для Junior-рівня.

3. Знання Web-технологій
3.1. Розуміння концепції клієнт-серверної архітектури є обов'язковим для будь-якого рівня веб-розробки.
3.2. Базова структура HTML-документа є необхідною, хоча глибоке знання HTML не є критичним для Junior PHP Developer.
3.3. Основні CSS-властивості та селектори є базовою компетенцією. Повне знання всіх властивостей не є обов'язковим.
3.4. Розуміння роботи API є обов'язковим. Рекомендується також перевіряти знання авторизації та базове розуміння API-документації (наприклад, Swagger/OpenAPI).
3.5. Робота з HTTP-запитами є базовою компетенцією. Окремо слід враховувати питання file upload як частину цього блоку.
3.7. Розуміння сесій та cookies є обов'язковим, включно з розрізненням даних, які слід зберігати у cookies та у sessions.

4. Робота з базами даних
4.1. Робота з MySQL та PDO є обов'язковою компетенцією для Junior PHP Developer.
4.2. Знання SQL є важливим, однак питання оптимізації та навантаження мають переважно теоретичний характер для цього рівня.
4.3. Використання PhpMyAdmin може бути корисним на старті, однак для професійної розробки бажано орієнтуватися на більш сучасні інструменти адміністрування баз даних.

5. Розширені знання PHP
5.3. Робота з датами та часом є важливою. Рекомендується поєднувати це питання з практичними сценаріями збереження даних у базі (наприклад, дата народження або дата транзакції).
5.4. Регулярні вирази є важливою темою, однак для Junior-рівня достатньо розуміння їх призначення та вміння користуватися документацією або онлайн-інструментами.
5.5. Обробка помилок та винятків вимагає розуміння не лише «як», а й «навіщо» — зокрема, чому недостатньо використовувати лише Error з кастомним текстом.
5.6. Основи безпеки (SQL injection, XSS, CSRF, password hashing) є критично важливими. Очікується поверхневе розуміння та базова практична обізнаність без глибокої експертизи.
5.7. Основи тестування (PHPUnit) є бажаною навичкою. Часто це те, чому Junior-розробники навчаються під час роботи.
5.8. Достатньо базового знання хоча б однієї CMS (WordPress, Magento або OpenCart).
5.9. Розуміння хоча б одного PHP-фреймворку (Laravel або Symfony) є стандартом для сучасної розробки.

6. Інструментарій розробника
6.1. Git є обов'язковим інструментом. Очікується базове володіння (pull, push, merge та робота з гілками за прийнятим workflow).
6.4. Базові навички Linux є необхідними. Щодо Docker — достатньо вміння працювати з готовими образами та повторювати типові сценарії використання.
6.5. Базове налаштування середовища розробки є необхідним. Також очікується вміння працювати з SSH (логін/пароль або ключі), окрім FTP/FTPS.

WEBSPARK

4.2 Знання мови SQL - пункт "Володіє навичками оптимізації запитів за допомогою індексів та розуміє, як уникати надмірного навантаження на базу даних." виділити окремо, так як ця компетенція має нижчу оцінку на рівень junior. Тобто, якщо "Здатний виконувати основні SQL-запити" має оцінку 3, то отпимізації я поставив би 2.

4.3 Адміністрування баз даних через веб-інтерфейс PhpMyAdmin - пункт "Володіє навичками створення резервних копій баз даних і відновлення даних з резервних копій для забезпечення безперервної роботи системи." більше відноситься до DevOps. Так как як займатись резервним копіюванням БД це задача не для junior.
Рекомендую замінити на пунк накшталт "Володіє навиками створення копій бази даних з використанням mysqldump інструмента".

Artjoker

1. Основи PHP. Базові знання з програмування
1.2. Оператори, цикли, інструкції мови та умовні конструкції є базовими знаннями. Важливо розуміти принципи керування потоком виконання програми та типові сценарії використання циклів (for, while, foreach). Окремо слід враховувати необхідність коректного виходу з циклів у разі потреби. При цьому «нескінченний цикл» сам по собі не є помилкою, а є конструктивним елементом, який використовується у серверних застосунках (наприклад, HTTP-сервери). Помилкою є лише відсутність логіки коректного завершення або виходу з такого циклу у відповідному контексті.
1.3. Функції є важливою частиною базових знань. Розуміння глобальних змінних є необхідним, однак їх практичне використання вважається небажаним підходом через ризики для підтримки та тестування коду.

2. Розуміння ООП
2.2. Інкапсуляція повинна розглядатися не як механізм «захисту даних», а як декларативний підхід до організації коду. Модифікатори доступу (private, protected, public) не забезпечують абсолютного захисту, оскільки доступ до них може бути отриманий через механізми на кшталт reflection. Основна мета інкапсуляції — визначення меж доступу та сигнал для інших розробників (або для майбутнього себе) про те, що внутрішня реалізація не призначена для зовнішнього використання.

3. Знання Web-технологій
3.7. Розуміння роботи із сесіями та cookies є обов'язковим. Кандидат повинен вміти створювати, зчитувати та видаляти cookies на стороні клієнта, а також налаштовувати їхній термін дії та область видимості. На стороні бекенду (PHP) важливим є розуміння роботи сесій та їхнього життєвого циклу.

4. Робота з базами даних
4.2. Знання SQL має включати базові операції (SELECT, INSERT, UPDATE, DELETE) та розуміння принципів побудови запитів. Оптимізація запитів, використання індексів та аналіз навантаження на базу даних належать до більш просунутого рівня і не є обов'язковими для Junior-позиції.
4.3. Використання PhpMyAdmin не є обов'язковою компетенцією. Натомість важливішим є вміння працювати з базами даних через термінал, зокрема використання інструментів на кшталт mysqldump, а також розуміння базових принципів роботи з MySQL у серверному середовищі.

5. Розширені знання PHP
5.7. Основи тестування (PHPUnit) залежать від практик конкретної компанії та проєкту. У більшості випадків це не є обов'язковою вимогою для Junior-рівня, але є суттєвою перевагою.
5.8. Робота з CMS (WordPress, Magento, OpenCart) є релевантною лише у випадку, якщо компанія використовує відповідні системи. В іншому випадку достатньо загального розуміння їх призначення.
5.9. Знання PHP-фреймворку (Laravel або Symfony) є залежним від технологічного стеку компанії. Якщо фреймворк використовується у проєкті — це обов'язкова вимога. Додатково це знання допомагає краще розуміти практичну реалізацію MVC-підходу.

6. Інструментарій розробника
6.5. Базове вміння встановлення PHP та налаштування локального середовища (наприклад XAMPP/WAMP/MAMP) не є обов' язковим. Натомість важливішими є навички роботи з серверним середовищем Linux, встановлення PHP та MySQL на сервері, а також базове розуміння принципів хостингу. Обов'язковою компетенцією є вміння працювати з SSH (ключі, авторизація, базові команди), оскільки це є стандартом у реальному продакшн-середовищі.

AltexSoft

ДОДАНО КОМПЕТНТНІСТЬ
6.6 Розуміння хмарних технологій.
Має розуміння сервісів популярних cloud-провайдерів (AWS, Azure, Google Cloud). Знає призначення та функції найбільш поширених сервісів (Lambda, SQS, SNS, EC2 тощо). Вміє працювати з цими сервісами, користувався офіційними SDK.