Компетентності рівня junior для JavaScript	Загальна оцінка компетентності (max. 27)	Компанії								
		ArtJoker	G5 Entertainment	SoftServe	Viseven	Knubisoft	CHI Software	Intellias	Telesens	SevenPro
1. Знання web-технологій, HTML, CSS										
1.1 Загальні питання веб технологій Розуміє клієнт-серверну архітектуру 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. Вміє використовувати cookie, LocalStorage для зберігання даних користувача	**23**	2	2	3	2	3	2	3	3	3
1.2 Кросбраузерність, адаптивність, mobile-first підхід, Accessibility. Pixel Perfect Розуміє різницю між різними веббраузерами, їх особливості у відображенні та інтерпретації вебсторінок. Знає сучасні стандарти та рекомендації веброзробки для забезпечення кросбраузерної сумісності. Вміє тестувати та налагоджувати вебсторінки у різних браузерах, щоб виявляти та виправляти проблеми сумісності. Має навички тестування адаптивності вебсторінок на різних пристроях та екранах за допомогою інструментів розробника браузера або мобільних емуляторів. Розуміє основні принципи налагодження адаптивності та вміє виправляти проблеми, такі як неправильне відображення, накладання чи обрізання елементів на різних пристроях. Здатний налаштовувати медіазапити CSS. Розуміє концепції Mobile First підходу, принципи побудови простої навігації, роботи з адаптивними зображеннями та швидким завантаженням сторінок. Знає основні принципи та рекомендації Web Content Accessibility Guidelines (WCAG) для створення доступних вебсторінок, вміє використовувати семантичні елементи HTML для забезпечення структурованості та доступності контенту.	**21**	3	1	3	2	3	2	3	1	3
1.3 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.	**26**	3	2	3	3	3	3	3	3	3
1.4 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	**24**	3	1	3	3	3	3	3	2	3
1.5 Знання одного з markup framework (Bootstrap CSS / Compass CSS / Foundation CSS Material UI / Semantic UI / Знайомство с LESS, SASS) Вміє використовувати готові компоненти та класи фреймворку для створення структури та стилів вебсторінки. Здатний налаштовувати та перевизначати стилі фреймворку відповідно до вимог проєкту Вміє компілювати та інтегрувати LESS або SASS у робочий процес розробки. Має навички використання змінних, міксинів та функцій препроцесорів для більш ефективної та модульної розробки стилів	**12**	1	2	2	0	2	1	0	3	1
1.6 Робота з DOM, Events Знає основи організації та подання елементів та структури вебсторінки у вигляді дерева об'єктів. Ідентифікує різних типів вузлів DOM-дерева. Вміє виконувати пошук та маніпулювати DOM-елементами за допомогою вбудованих методів, працювати з координатами елементів, отримувати та змінювати позиції елементів на сторінці. Має навички керування атрибутами та вмістом DOM-елементів. Розуміє процес призначення обробників подій елементам, механізм спливання та перехоплення подій, використовує делегування подій для ефективної обробки подій на батьківських елементах. Ідентифікує та використовує обробку подій від різних пристроїв (події миші, клавіатури, прокрутки)	**24**	3	2	3	3	3	3	2	2	3
2. Теоретичний блок. Розуміння ООП										

2.1 Об'єкти Знає принципи використання об'єктів для організації та структурування даних та функціональності у програмі. Розуміє основні концепції ООП (інкапсуляція, спадкування та поліморфізм) Ідентифікує поняття та концепцію поліморфізму та його роль в ООП. Вміє створювати та використовувати поліморфні методи, які можуть виконувати різні дії залежно від типу об'єкта.	**25**	3	3	3	2	3	3	3	3	2
2.2 Класи, конструктори, властивості, модифікатори доступу Ідентифікує поняття класу та його роль в ООП. Знає синтаксис класів у JavaScript та їх використання для створення об'єктів. Вміє створювати ієрархії класів за допомогою наслідування та з використанням конструкторів. Розуміє відмінності між класами та функціями-конструкторами. Розуміє механізм успадкування властивостей та методів між класами. Усвідомлює поняття приватних властивостей та їх використання для приховування даних від зовнішнього доступу. Використовує різні підходи до реалізації приватних властивостей JavaScript. Працює з аксессорами. Розуміє статичні властивості та методи класу та їх ролі. Вміє використовувати статичні властивості та методи для роботи із загальними даними та функціональністю класу.	**22**	1	3	3	2	3	3	3	2	2
2.3 Прототипне наслідування - властивість proto. Функціональне наслідування Розуміє прототипне наслідування та його відмінності від класичного наслідування. Використовує властивість proto та усвідомлює її роль в ланцюжку прототипів об'єктів. Вміє використовувати прототипи для успадкування властивостей та методів з інших об'єктів Розуміє призначення функцій-конструкторів та їх використання для створення об'єктів з успадкованими властивостями та методами.	**17**	2	3	3 / 0	1	3	3	0	2	1
3. Основи JavaScript										
3.1 Змінні та типи даних. Масиви та Рядки Знає різні типи даних у JavaScript (числа, рядки, булеві значення, об'єкти та масиви), здатний використовувати їх для зберігання даних, усвідомлює правила приведення типів. Розрізняє статичну та динамічну типізацію. Ідентифікує оператори var, let та const, локальні та глобальні змінні, розуміє механізм hoisting Вміє працювати з рядками, масивами, знає особливості використання вбудованих функцій для цього, розуміє поняття "об'єкт-обгортка". Знає особливі колекції Map та Set, особливості та доречність їх використання.	**27**	3	3	3	3	3	3	3	3	3
3.2 Оператори та цикли, умовні конструкції Має навички роботи з логічними, арифметичними, бітовими та операторами присвоювання, ідентифікує їх пріоритет виконання. Будує логічну структуру програми з використанням умовних операторів, операторів розгалуження та циклів, розрізняє можливості використання continue та break операторів. Працює з перебираючими методами для масивів та об'єктів. Розуміє особливості використання Symbol (ітератор)	**26**	3	3	3	2	3	3	3	3	3
3.3 Функції Використовує функції при побудові програми мовою JS, ідентифікує особливості використання Function declaration, Function Expression, Arrow function, функції-конструктора, IIFE. Вміє працювати з параметрами функції, об'єктом arguments, використовує rest та spread оператори, деструктуризацію. Має досвід створення рекурсивних функцій, використання функцій-колбеків, розуміє механізми чейнінгу та карування функцій.	**25**	3	3	3	2	3	3	3	2	3
3.4 Date. Обробка дати та часу Має навички роботи зі спеціальними об'єктами Date, обробкою даних у форматі дати/часу. Розуміє принципи роботи з таймерами та інтервалами, періодичним викликом функцій.	**18**	2	0 / 3	2	1	3	2	0	3	3
4. Розширені знання JavaScript.										
4.1 Контекст виклику. Замикання Розрізняє особливості роботи з контекстом (this). Практикує використання вбудованих функцій для запобігання втрати контексту (call, apply, bind). Розуміє та використовує замикання, ідентифікує область видимості функції, лексичну видимість. Знає особливості hoisting у функцій та змінних Розуміє структури лексичного середовища (lexical environment) і його ролі в замиканнях. Здатний використовувати замикання для збереження стану та доступу до змінних.	**26**	3	3	3	3	3	3	3	2	3
4.2 Асинхронність в JS Розуміє механізм Event Loop, особливості роботи з чергами мікро- та макро-тасків. Ідентифікує різні підходи роботи з асинхронним кодом - Callback, Promises, Async/await. Розуміє принцип роботи та особливості використання функцій-генераторів. Використовує асинхронних запитів до сервера з використанням технології AJAX (Asynchronous JavaScript and XML) для отримання або відправлення даних без перезавантаження сторінки. Здійснює завантаження та обробку файлів асинхронно, використовуючи об'єкт FileReader або fetch API Вміє здійснювати обробку помилок, що виникають при виконанні асинхронного коду, зокрема використовує блок try/catch для перехоплення та обробки помилок.	**22**	2	2	3	2	3	3	2	2	3
4.3 Регулярні вирази Знає особливості використання регулярних виразів для пошуку та зіставлення текстових шаблонів, методи роботи з ними. Працює зі структурою шаблонів, метасимволами, модифікаторами та флагами, дужками та квантифікаторами.	**15**	2	1	2	0	2	1	3	2	2
4.4 Робота с webpack та JavaScript-модулями Розуміє стандарт модулів ES6, концепцію модулів в JavaScript, організацію коду з "export" та "import" Вміє використовувати webpack для збирання та упаковування JavaScript-додатків. Працює з конфігурацією webpack для збірки розробчої або продакшн версій, обробки різних типів файлів (наприклад, CSS, зображень), налаштування режиму розробки. Використовує завантажувачі (loaders) у webpack для обробки різних типів файлів. Вміє використовувати пакетний менеджер (наприклад, npm або Yarn) для установки та керування залежностями проєкту. Здатний встановлювати та використовувати зовнішні пакети, які можуть бути імпортовані в проєкт через модульну систему.	**17**	2	2	2	2	2	2	2	1	2

4.5 Принципи роботи DRY, YAGNI, KISS, Separation of Concerns, Використовує рефакторинг для поліпшення коду згідно з принципами DRY та YAGNI. Розуміє необхідність розробки за допомогою функцій, класів та модулів для уникнення дублювання коду. Уникає надмірного написання коду або включення функціональності, яка на цей момент не потрібна. Фокусується на поточних вимогах та функціоналісті проєкту та уникає зайвого розширення або узагальнення Дотримується принципів розділення функцій і обов'язків між різними компонентами системи для поліпшення читабельності, підтримки і можливості повторного використання.	**14**	0	3	1	2	1	2	0	2	3
4.6 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. Володіє навичками читання та створення даних у форматах JSON і XML, ідентифікує переваги та недоліки кожного формату і вміє обирати відповідний формат для конкретного випадку використання. Здатний використовувати різні методи автентифікації, такі як токени, сесії та OAuth. Вміє налаштовувати та використовувати механізми автентифікації та авторизації у REST-застосунках. Розуміє рівні доступу та особливості використання ролей користувачів для обмеження доступу до ресурсів.	**20**	3	2	3	1	3	2	2	2	2 / 3
4.7 Концепції MVC та MVT Вміє структурувати додаток за концепцією MVC / MVT та встановлювати взаємозв'язки між компонентами. Усвідомлює процес передачі даних між моделлю, представленням та контролером (шаблоном). Розуміє принцип одностороннього потоку даних, де модель оновлюється контролером, а представлення отримує оновлені дані з моделі Знає принципи SOLID (Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion) та їх використання для покращення архітектури додатка, зокрема моделі, представлення та контролера або шаблону. Розуміє інші архітектурні шаблони, які можуть бути використані разом з MVC або MVT, наприклад, MVP (Model-View-Presenter) або MVVM (Model-View-ViewModel). Вміє розробляти добре структурований та підтримуваний код, використовуючи концепції MVC або MVT та інші принципи проєктування	**14**	0	2	3	1	1	1	2	2	2
4.8 Патерни розробки Знає про різні типи патернів проєктування, такі як структурні, поведінкові та породжувальні патерни. Розуміє основні принципи, які лежать в основі кожного патерну. Вміє впізнавати проблеми, для яких патерни проєктування є відповіддю, і використовувати патерни для їх розв'язання. Використовує популярні патерни розробки, такі як Singleton, Factory, Observer, Strategy, Decorator та інші. Вміє вибрати та використовувати відповідний патерн для конкретної ситуації розробки. Ідентифікує переваги та недоліки кожного патерну та здатний прийняти обґрунтоване рішення про їх використання.	**15**	0	2	2	1	3	1	2	2	2
5. Знання екосистеми JS. FrontEnd / Backend / FullStack										
5.1 Розуміння одного з фреймворків JavaScript (React, Angular або Vue.js), FrontEnd-розробка Знає основні концепції фреймворку, такі як компоненти, директиви та роутінг. Вміє створювати компоненти та працювати з їхнім життєвим циклом. Розуміє систему управління станом фреймворку та здатний використовувати стейт-менеджерів, таких як Redux, Vuex або NgRx. Використовує директиви та шаблони для маніпуляції DOM та створення динамічного інтерфейсу. Здатний застосовувати механізми роутінгу для навігації між сторінками та створення SPA (Single Page Applications). Розуміє підходи до обробки подій та взаємодії з користувачем у контексті фреймворку. Вміє працювати з API сервера та виконувати HTTP-запити з використанням фреймворку. Розуміє принципи реактивного програмування та використовує відповідні механізми фреймворка (Observables в Angular або React Hooks). Вміє розбиратися у документації фреймворку та використовувати ресурси спільноти для розв'язання проблем та отримання нових знань.	**21**	2	3	3	2	3	3	0	2	3
5.1.1 Вміє працювати з React Розуміє основні концепції React, такі як компоненти, властивості (props) та стан (state). Вміє створювати функціональні та класові компоненти та працювати з їхнім життєвим циклом, розуміє особливості використання синтаксису JSX. Використовує React Router для налаштування роутінгу та навігації між сторінками в односторінкових додатках. Розуміє принцип управління станом за допомогою локального стану (local state) та особливості використання React Hooks (таких як useState, useEffect) для роботи зі станом. Користується бібліотекою Redux для управління станом додатка та розширення його функціональності з допомогою middleware. Розуміє підхід "унідирекційного потоку даних" (Unidirectional Data Flow) та використовує React Context для передачі даних по дереву компонентів. Вміє працювати з зовнішніми бібліотеками та компонентами, такими як Material-UI або React Bootstrap, для швидкого розроблення інтерфейсу. Практикує використання інструментів для розробки (Create React App або Next.js) для підготовки та оптимізації проєкту.	**14**	1	2		0	3	3	3	2	
5.1.2 Вміє працювати з Angular Розуміє основні концепції Angular (модулі, компоненти, шаблони та сервіси). Вміє створювати компоненти та працювати з їхнім життєвим циклом, включаючи методи ngOnInit, ngOnDestroy тощо. Використовує RxJS для реактивного програмування та обробки асинхронних подій та запитів. Працює з шаблонами Angular, включаючи вирази, директиви, властивості (properties) та події (events). Використовує Angular Router для налаштування роутінгу та навігації між сторінками в додатках. Користується сервісами Angular для спільного використання даних та функціональності між компонентами. Застосовує декоратори Angular (@Component, @Injectable та @NgModule) для налаштування та розширення функціональності додатка. Розуміє особливості використання системи управління станом NgRx для ефективного керування станом додатка та уникнення проблем зі збереженням та синхронізацією даних. Знайомий з інструментами Angular CLI для швидкого створення проєкту, генерації компонентів, модулів та інших елементів аплікації.	**11**	1	2		0	0	3	3	2	

5.1.3 Вміє працювати з Vue.js Розуміє основні концепції Vue.js, такі як компоненти, директиви, обчислені властивості (computed properties) та слухачі подій (event listeners). Вміє створювати та використовувати компоненти Vue та працювати з їхнім життєвим циклом, включаючи методи mounted, updated та інші. Знає шаблони Vue (вирази, директиви та умовні рендеринги). Працює з директивами Vue для маніпуляції DOM та взаємодії з користувачем. Використовує Vue Router для налаштування роутінгу та навігації між сторінками в односторінкових додатках. Користується системою управління станом Vuex для керування та спільного використання стану додатка між компонентами. Розуміє особливості застосування Vue CLI для швидкого створення проєкту, генерації компонентів, модулів та інших елементів аплікації. Використовує механізм реактивності Vue.js для автоматичного оновлення компонентів при зміні даних. Знайомий з інструментами для розробки, такими як Vue Test Utils або Jest, для тестування компонентів та функціональності додатка, Vue DevTools для відлагодження (debugging) та аналізу додатка.	**13**	1	2		2	0	3	3	2	
5.2 Розуміння роботи з Browser API та Node.js (Core.js), Backend-розробка Вміє використовувати специфічну для браузера функціональність, маніпуляцію сторінками, роботу з подіями. Розуміє особливості роботи з файловою системою, мережевими операціями в Node.js, архітектуру та принципи роботи, включаючи його однопотокову та неблокуючу (non-blocking) природу Знає принципи роботи з асинхронним кодом та використання промісів (Promises) або async/await для керування асинхронними операціями. Використовує REST або GraphQL для розробки API та взаємодії з клієнтськими додатками. Практикує використання модулів та інструментів, що спрощують розробку на Node.js, наприклад, Express.js, Socket.io, Passport.js Застосовує пакетний менеджер npm для управління залежностями проєкту та установки необхідних модулів. Володіє принципами безпеки та автентифікації в бекенд-додатках, включаючи використання токенів або сесій. Вміє проводити тестування бекенд-додатків, включаючи юніт-тести та інтеграційні тести. Використовує інструменти для логування, моніторингу та налагодження бекенд-додатків, наприклад, з використанням логерів або інструментів для відлагодження.	**12**	0	2		2	0	1	3	1	3
5.3 Робота з базами даних SQL Має базові знання SQL syntax, вміє будувати запити (queries) для отримання інформації з бази даних, модифікації даних та виконання інших операцій. Вміє працювати з таблицями та схемами, розуміє зв'язки між таблицями. Здатний працювати з ключами та індексами для забезпечення ефективного доступу до даних. Розуміє типи з'єднань (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN) та може використовувати їх для об'єднання даних з різних таблиць та отримання з'єднаних даних Може будувати та виконувати SELECT-запити для отримання певних даних з таблиць та фільтрацію даних на основі певних умов, здійснювати їх агрегацію та групування Вміє працювати зі схемами баз даних, встановлювати права доступу користувачів тощо	**11**	0	1	3	1	0	0	2	1	3
5.4 Робота з базами даних NoSQL Знає різні інструменти та бібліотеки для роботи з NoSQL базами даних, такі як MongoDB, Cassandra, Redis, Couchbase Розуміє різницю між ACID (Atomicity, Consistency, Isolation, Durability) транзакціями, що характерні для реляційних баз даних, та BASE (Basically Available, Soft state, Eventually consistent) підходом, що використовується в NoSQL базах Вміє працювати зі специфічними структурами даних, які забезпечуються NoSQL базами, такими як дерева, графи, документи тощо Має навички тестування взаємодії NoSQL баз даних з іншими компонентами, такими як бекенд, фронтенд, API	**8**		1	3	1	0	0	2	1	
5.5 Розуміння роботи jQuery UI Знає основні компоненти та функціональність jQuery UI, такі як діалогові вікна, перетягування (drag and drop), сортування, автодоповнення. Вміє використовувати різні ефекти та анімацію для покращення користувацького досвіду на веб-сторінці. Застосовує методи та події jQuery UI для взаємодії з елементами на сторінці та виконання певних дій	**3**	0	0	0	0	0	1	0	1	1
5.6 Розуміння роботи JS Testing frameworks (Jest, Jasmine,..) Розуміє роль тестування в процесі розробки програмного забезпечення. Вміє налаштовувати та використовувати різні JS тестові фреймворки, такі як Jest, Jasmine, Mocha, Chai та інші. Знайомий з основними концепціями тестування (юніт-тести, інтеграційні тести, тести з використанням розширення (mocking) тощо). Вміє писати ефективні й надійні тести для перевірки функціональності JavaScript-коду.	**13**	0	1	2	1	3	1	2	1	2
5.7 Розуміння роботи JS Template Engine. Розуміє ролі та призначення шаблонних двигунів (template engines) у веброзробці. Використовує популярні шаблонні JS двигуни, такі як Handlebars, Mustache, EJS. Створює шаблони для генерації HTML-коду або іншого формату даних на стороні клієнта або сервера. Ідентифікує принципи роботи шаблонів, включаючи вбудовані вирази, умови, цикли та інші конструкції шаблонів.	**9**	1	1	2	0	2	1	0	1	1
6. Інструментарій розробника										
6.1 Системи контролю версій - GIT Розуміє git flow - методологію роботи з гілками для ефективного управління процесом розробки. 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.	**24**	3	2	3	2	3	3	2	3	3

6.2 Навички використання IDE Вміє працювати з популярними IDE, такими як Visual Studio Code, IntelliJ IDEA, Eclipse, або WebStorm. 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.	**17**	2	1	3	1	3	2	1	3	1
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.	**13**	2	0	2	1	2	2	0	2	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.	**10**	1	0	2	0	3	1	0	1	2
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.	**18**	3	1	3	0	3	2	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.	**15**	3	2	2	0	2	2	1	1	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.	**16**	3	2	2	0	2	2	1	2	2
7.4 Професійна термінологія для JS Знає та розуміє специфічну термінологію, що використовується в контексті JavaScript-розробки Розуміє та використовує специфічні терміни, патерни, алгоритми, фреймворки, бібліотеки та інші інструменти, які використовуються в JavaScript-розробці.	**23**	3	3	3	2	3	2	2	3	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.	**12**	3	1	1	0	1	2	0	2	2

Коментарі до таблиці

Оцінки компетентностей вказані станом на 2023/2024 рр., по шкалі від 0 до 3.

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

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

«1» - некритична компетентність, якою можна знехтувати.

«0» - неактуальна компетентність.

Зауваження від компаній

ArtJoker

2.1 - Як мінімум потрібно розуміти. Що таке об'єкт, які об'єекти бувають, які властивості бувають, дескриптори властивостей і т.д.

5.1 - Як мінімум базові основи одного з перелікованних, потрібно мати.

5.1.1-5.1.3 - Залежить від того, потрібно буде розробляти проекти на реакті/ангулярі/вью, чи ні

5.2 - Для JS розробника не потрібно

5.5 - Тільки якщо на проекті уже використовується ця біліотека

5.7 - Вони не складні, зайвим не буде, на співбесіди малоймовірно що запитають

7.1-7.5 - Якщо, потрібно спілкуватися з замовником, та замовник не знає Української мови, то так.
Якщо задачі розробнику поступають через PM і розробник не комуніцирує з англомовним клієнтом. То знання мови немає жодного значення
Для розвитку особистого 100% потрібно знати Англійську, внезалежності не від чого.

G5 Entertainment

компетентності оцінено для фронтенд розробника

1.2 - Це компетенція верстальника, а не розробника. Але треба розуміти, що це таке.

1.3 - Це компетенція верстальника, а не розробника. Але розробнику треба розуміти що таке структура HTML-документа, знати основні HTML-теги, розуміє їх призначення та особливості.

1.4 - Це компетенція верстальника, а не розробника. Але треба розуміти, що це таке.

1.5 - Це не критична компетентність,але буде великим плюсом якщо знатиме це.
Можна об'єднати пункти 1.2, 1.4, 1.5 на один загальний, так як це компетенції скоріше для верстальника ніж розробника.

3.4 - З датою та часом - 0 Робота з інтервалами та таймерами - 3.

SoftServe

Окрім JavaScript, набирає оберти TypeScript. Для побудови нових великих проєктів, перевагу віддають саме TypeScript замість JavaScript.

2.3 - прототипне наслідування 3, функціональне наслідування 0

5.1.1 - React найбільш популярний серед трьох зазначених

5.4 - MongoDB обов'язково

Viseven

1.5 - Ми використовуємо тільки SASS і тільки базові можливості, тому на цей пункт не звертаю уваги, особливо для junior-рівня

4.2 - Найбільшу увагу слід приділити підходам до роботи з асинхронним кодом, більшість їх не розуміє

4.3 - Задачі з регулярними виразами не так часто трапляються

CHI Software

4.4 - Усе, окрім вебпаку - мастхевні знання.

4.5 - Треба не лише мати розуміння цих паттернів, але й мати достатньо досвіду використання.

4.6 - Повинен мати знання про будування запитів, використання HTTP методів. Розуміє як працювати з JSON.

5.1.1 - 5.1.3 - Мастхев, якщо фремворк є профільним.

5.7 - У разі знання одного з ключових фреймворків - джуніор зіштовхується з якоюсь варіацією темплейтингу, тож окремо це вивчати не обов'язково.

6.3 - Праця з гітом, навігація по файловій системі - мастхев.

Intellias

1.5 - Не потрібно для джуніорів

1.3 - 1.4 - 1.6 можна скомбінувати в один

2.3 - Не актуально для джуніорів

3.3 - IIFE - майже не використовується на сучасних проектах.

3.4 - Не потрібно для джуніорів

4.4 - Можна додати "розуміння webpack module federation". Також перейменвувати "Робота с webpack та JavaScript-модулями" в "Build tools and javascript-modules"

4.5 - Не потрібно для джуніора

4.7 - MVT зайве.

5.5 - вже майже ніде не використовується

5.7 - зайве. з нашого досвіду таких проектів дуже мало

6.2 - Не так критично для джуніора

6.3, 6.4 - Не потрібно джуніору

7.1 - Очікуваний рівень англійської B1

7.2,7.3,7.5 - Не критично для джуніора

7.4 - Добре розуміти та оперувати термінами англійською мовою

Telesens

1.2 - ці знання будуть однозначно плюсом проте такий великий ліст навиків зазвичай зустрічається на рівні Junior вкрай рідко. Є просто загальні поняття

2.2, 2.3 - на цьому рівні ми не отримаємо глибоких знань

3.3 - не всі поняття кандидат з рівнем Junior зможе пояснити - Function declaration, Function Expression тут думаю будуть блокери в цьому контексті

6.3 - гарно якщо розуміється в цьому

SevenPro

4.6 - 3 - для backend розробника, 2 - для frontend розробника

5.1 - Підпункти неможливо оцінити, так як це залежить від вимог проекту

5.3,5.4 - Оцінка наведена в рамках знань з баз даних. Вимоги, щодо специфічних можливостей окремої бази, можна наводити тільки на основі вимог проекту

6.3, 6.4 - Для backend розробника - 3