**МОНТАЖНИЙ ПЛАН: F&B**

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва курсу** | **IT-Box: цікаві рішення для початківців** | **Кількість мікролекцій (шт)** |  |
| **Назва тематичної лекції** | **F&B** | **Тривалість повної лекції (хв)** |  |
| **Номер лекції** |  | **Тривалість мікролекцій (хв)** |  |

**МІКРОЛЕКЦІЯ №1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TEXT  ITEM № | ТЕКСТ / ТРИВАЛІСТЬ ДЕМОНСТРАЦІЇ | ТИП | Що має бути показано | Коли має бути показано | | Як має бути показано |
| 1 | Привіт, друзі. Я Федоренко Олексій і сьогодні ми познайомимося з вебом, однією з найпоширеніших галузей в програмуванні. В цьому уроці я розкажу як і чому вчити веб.  Хотілося б почати з того що звичайні веб програмісти гори грошей не заробляють, але в принципі, їх труд непогано оцінюється. Тому, після опанування бази, я раджу вам спробувати себе в невеликі компанії, щоб набратися більше досвіду і підучитися у інших розробників. А після такого досвіду можна йти на фріланс, в більш велику компанію, або навіть створювати свій проект. Щоб з максимальною користю інвестувати свій час, при виборі компанії звертайте увагу на те, чи є там з система грейдингу і чи зацікавлений роботодавець в особистісному зростанні співробітників. В такому випадку, у вас буде хороший тімлід, працюючи з яким ви зможете розвиватися. Тому якщо ви гарно опануєте програмування і додаткові навички, то зможете легко влаштуватися в відому компанію, тим більше що закордонні гіганти в технологічній сфері роблять все можливе, щоб заманити професіоналів до себе в офіси.  Якщо ж вас не дуже надихає можливість працювати на компанію, і хочете подорожувати по світу, працювати дома і не прив’язуватися до графіку то веб дає можливість працювати на фрілансі, однак там рівень навичок для входу набагато вищий, чим в компаніях. Також, скоріше за все потрібно буде працювати навіть більше ніж на компанію, щоб отримувати схожу зарплату, але плюсом є те, що не обов’язково мати офіційну освіту. Всі міфи про багатих фрілансерів йдуть скоріше від рідкісних виключень або успішних стартапів. Також з фрілансом може виникнути ситуація коли людині просто не вистачає соціуму, адже працюючи в компанії ти оточений безліччю співробітників і можеш з ними спілкуватися в живу.  Стаючи веб програмістом, це відкриває для вас неймовірний простір для творчості. Тому, є варіант для створення стартапу, Білом Гейтсом чи Марком Цукербергом ви навряд чи станете, однак заробляти на достойне життя доволі реально, хоча цей варіант найризикованіший і буде потребувати від вас найбільшої відданості і професіоналізму в дуже широкій сфері. Адже на перших етапах мало в кого є гроші на працівників. В будь-якому випадку вам доведеться зробити вибір, якого шляху заробітку дотримуватися. | Vector img | Image result for money | Red text | Cross out the img | |
| Vector img |  | Purple text |  | |
| Vector img |  | Light red text |  | |
| Vector img |  | Orange text |  | |
| Vector img |  | Tan text |  | |
| Text | * Офісний планктон * Фріланс * Стартап | Dark blue text |  | |
| 2 | Деякі кажуть, що програмування дуже просто вивчити і залиште це на останній день, ви будете мотивовані, і все встигнете зробити. Але це не так, тому поки у вас є пристрасть навчитися якому небудь ремеслу, будь то програмування або спортивне поїдання сосисок - взагалі не важливо. Направте цей імпульс в русло самоорганізації. Я маю на увазі, що в першу чергу вам потрібно скласти план навчання, поки мотиваційний прихід вас не відпустив. Що стосується програмування, то можете скористатися готовим планом, який вам пропонують досвідчені бородаті дядьки. Просто берете книгу і ставите собі за мету кожен день скурювати по розділу. Особливо великі і складні глави розтягуйте на кілька днів. Дуже важливо не починати вивчення наступного розділу, поки не освоїли попередній, а то потім прогалини в знаннях будуть більші ніж в Уиллема Дефо між зубами. Також не забувайте відпрацьовувати кожен момент який ви вивчили, щоб закріпити матеріал. Ось вам найпростіший план. Також, не покладайтеся сліпо на курси, які обіцяють зробити з вас відразу Middlа, без самостійної наполегливої роботи і практики результату не буде, ви просто витратите гроші.  Чим далі ви будете заходити, тим менше буде залишатися мотивації. Згодом її буде замінювати дисципліна. Виділіть собі конкретний час в графіку для занять. Наприклад, приходячи додому зі школи \ Галери \ універу відразу беріться за програмування на годинку \ дві \ три, це як ви вже самі вирішите. І потім сміливо топайте займатися своїми справами, там з пацанами в Падик сємки по клацати, та що завгодно.  Інколи навіть розуміння і дисципліни не достатньо щоб кожен день займатися справою, тоді потрібно вчитися обманювати свій мозок і змушувати його працювати. Для цього я використовую правило 5 секунд. Тобто мені потрібно обов'язково займатися справою тільки на протязі 5 секунд, якщо я після цього проміжку не хочу продовжувати, я цього не роблю. Наш мозок дуже не любить не закінчену роботу, тому скоріше за все ви продовжите і доведете справу до кінця. Ключовий момент полягає в тому, що б ви не пропускали ні дня і правильно спланували час навчання, що б воно не приносило вам занадто потужний дискомфорт. Займайтеся, нехай по трошки, але кожен день. Частота важливіше обсягу. | Vector img |  | Red text | | Cross out the img |
| Vector img |  | Dark blue text | |  |
| Vector img |  | Purple text | |  |
| Vector img |  | Aqua text | |  |
| Vector img |  | Tan text | | One by one |
| Vector img |  | Orange text | |  |
| 3 | Ну і, звичайно, вам навряд чи вдасться протриматися довго, якщо ви не поставите собі конкретну мету. Наприклад, "я хочу стати топ хакером, зламувати все що погано лежить, і заснувати університет імені мене". Переконайтеся, що ця мета для вас дійсно важлива, і її можна реалізувати, і при чому в не дуже довгий проміжок часу. Але тут потрібно зробити такий відступ, що якщо вас не надихає витратити близько 2000 годин свого життя щоб почати непогано розбиратися в Web Development +- на рівні мідла, то дуже серйозно подумайте, чи потрібно воно вам. Тим більше якщо ви в школі чи університеті, то навряд чи ви зможете багато часу приділяти програмуванню. І у вас цей шлях може зайняти близько 4-5 років, як це було в мене. І це в тому випадку, що будете серйозно працювати на вихідних і канікулах, витрачаючи свій вільний час. Не обов'язково кожен повинен вміти програмувати. І якщо ваш мотив заснований на тому щоб зрубити бабла на швидку руку, або так мама сказала. То скоріше за все програмування не для вас. Якщо ж ви не знаєте чи хочете ним займатися, то спробуйте і одразу зрозумієте. Якщо вас все влаштовує і програмувати вже й не дуже хочеться, то просто не витрачайте час. Будьте впевнені в тому, що програмування це саме те, що вам потрібно. Також для того щоб стати професіоналом, не потрібно бути генієм, саме головне це послідовність і старанність. Тому, щоб не втратити бажання працювати не потрібно постійно порівнювати себе з іншими, порівнюйте свої теперішні досягнення з своїми попередніми. Тому пам'ятайте, Нео був обраним, але навіть у нього не вийшло з першого разу перестрибнути з одного даху на інший. Помилки загартовують - пишіть одну і ту ж програму, переверстуєте сайт до тих пір, поки не вийде ідеально. Якщо ж ви вже другий день підряд працюєте над помилкою і не можете її вирішити я вам дам таку пораду - ідіть поспіть. Наш мозок перестає продуктивно працювати, якщо ми перевантажуємо його роботою. | Vector img |  | Red text | |  |
| Vector img |  | Green text | | One by one |
| Vector img |  | Orange text | | Cross out the img |
| Vector img |  | Dark blue text | |  |
| 4 | Становлення гарного програміста складається не тільки з вміння прогати, але й з допоміжних навичок. Зокрема я кажу про англійську мову. І про навички спілкування з людьми. Англійська відкриє для вас нові можливості, з пошуку інформації, все-таки документація для мов програмування в оригіналі писалася англійською, а все інше це просто адаптації. Крім того Інгліш дасть вам можливість працювати з іноземними клієнтами та компаніями. Що відкриває більш прибутковий і менш проблемний ринок, ніж робота з товаришами в СНД.  З приводу "навички спілкування з людьми", її чомусь часто не беруть до уваги, а деякі взагалі уявляють програмістів соціофобами. Хоча програмістам досить часто доводиться спілкуватися з людьми, а в особливості на фрілансі, де кожен раз спілкуєшся з новим клієнтом. На тій самій співбесіді з кандидатів, в яких однаковий рівень знань, виберуть того, який краще володіє навиками комунікації. Так що чувачелікі, цей навик потрібен, що б ви змогли реалізувати себе як програміста. А то смішно вийде, якщо ти найкращий програміст в цьому краю землі, а з тобою ніхто не хоче співпрацювати, бо ти розучився спілкуватися і можеш відповідати тільки командами з комп’ютера.  Як не дивно, кудись йти веселіше якщо знати дорогу. Саме тому у наступних відео ми подивимося що ж потрібно веб розробнику в 2018 році. А на цьому все. З вами був Федоренко Олексій, удачі вам у становленні веб розробником. | Vector img | Image result for speaking | Dark blue text | |  |
| Vector img | Related image | Red text | |  |
| Vector img |  | Orange text | |  |
| Vector img |  | Green text | |  |

**МІКРОЛЕКЦІЯ №2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEXT  ITEM № | ТЕКСТ / ТРИВАЛІСТЬ ДЕМОНСТРАЦІЇ | ТИП | Що має бути показано | Коли має бути показано | Як має бути показано |
| 1 | В цьому відео я розкажу про технології програмування саме для web frontend, які на мою думку можна вивчати в 2018 році. Також дам поради на рахунок того де ці технології використовуються і що краще вивчати новачку.  Вивчення frontend легше у відношенні до фундаментальних процесів, ніж в інших частинах комп'ютерних науках, таких як машинне навчання або системне програмування. Також для багатьох буде плюсом те, що незнання математики - це невелика перешкода для frontend розробника, хоча я вважаю її знання непоганим бонусом, бо математика вчить послідовно мислити. Але не вірте людям, які кажуть що frontend це просто. Щоб стати справжнім професіоналом в frontend, потрібно витратити насправді дуже багато часу. Адже програмувати сайти потрібно правильно, зберігаючи основні норми, і так щоб інші програмісти могли з вами працювати. Тим більше писати все з нуля не дуже ефективно і прибутково, тому доведеться освоїти безліч різних бібліотек, фреймворків та додаткових систем для покращення ефективності. | Vector img |  | Green text |  |
| Vector img |  | Red text | Make a bonus overlay after the green word |
| Vector img |  | Dark blue text | Cross out the img |
| 2 | Ми завжди починаємо з основ. HTML є основою веб сайту, якщо порівнювати з біологією людини, HTML - це кістки та структура. Це скелет людського тіла. Сам по собі, він дійсно не може багато чого зробити, але він визначає структуру. HTML це не мова програмування, але це інструмент, який повинен опанувати кожен веб-розробник.  HTML розшифровується як (Hyper Text Markup Language), або українською - Мова розмітки гіпертекстових документів. Він в основному являє собою набір блоків, які ми можемо використовувати для визначення структури нашого веб-сайту.  Було б непогано почати вивчення HTML з версії 4. Вона простіша, там менше блоків, але й менше можливостей. З часом все рівно доведеться перейти до версії HTML5, яка являється останньою на сьогоднішній день. Не переживайте, все що працювало в  HTML4 буде працювати і в версії 5. HTML5 просто тепер включає нові теги, що описують частини документа. Наприклад не <div id=”header”>, a <header>. Також додалися нові елементи форми, додалася підтримка аудіо та відео. І мабуть найголовніша зміна це тег canvas, що дозволяє побудувати графіку в інтерактивному режимі. Це дасть змогу навіть робити ігри.  World Wide Web Consortium (W3C) є основною організацією міжнародних стандартів для веб сторінок. Тому потрібно одразу вчитися як писати код правильно, слідуючи їх стандартам. Використовуючи W3C валідатор можна перевіряти HTML або CSS-код сайту і отримувати звіт про помилки в коді, або дізнаватися де код не відповідає стандартам. Також після вивчення курсу HTML ви вже повинні знати основні блоки, або по іншому теги, які параметри вони мають, знати базову структуру сайту, що таке клас, ід. Допомогую в написанні коду може бути Emmet, що являється набором плагінів для текстових редакторів, які дозволяють здійснювати високошвидкісне кодування та редагування формату HTML. | Vector img |  | Red text |  |
| Text | HTML - Hyper Text Markup Language | Dark blue text |  |
| Text | HTML4 -> HTML5 | Green text |  |
| Text | <div id=”header”> -> <header> | Tan text |  |
| Text | <canvas></canvas> | Purple text |  |
| Text | * W3C - World Wide Web Consortium * W3C validator | Orange text | One by one. Second after word validator |
| Text | * Main tags * Structure * Class * Id * Emmet | Aqua text | One by one after mentioning |
| 3 | Після створення основи сайту ми можемо почати роботу з CSS, що розшифровується як (Cascading Style Sheets) і звучить українською як Каскадні таблиці стилів. CSS несе відповідальність за візуальну частину. Дозволяючи нам програмувати зовнішній вигляд елементів, утворених завдяки тегам HTML. Ми можемо визначити розміри елементів, кольори, фони та багато іншого.  Повертаючись до моєї біологічної метафори, ми могли б сказати, що CSS характеризує людину. Він розповідає нам про колір шкіри, очі, висоту тощо. На етапі вивчення CSS ми отримуємо людину, яка ще нічого не робить, але має тіло (HTML) і зовнішній вигляд (CSS).  Вивчаючи CSS ви обов’язково повинні розібратися в роботі Box model. Ця структура, по суті, являє собою багато шарів, кожен з який послідовно огортає кожен елемент HTML.  Весь сайт побудований використовуючи спеціальні системи відношення елементів між собою. Grid system від Bootstrap та Flexbox являються одними з таких систем які складаються з набору вимірювань, які веб дизайнер може використовувати для вирівнювання та розміщення об'єктів у заданому форматі. Хоча й існують такі стандарти, немає строгих правил щодо розміщення елементів. | Text | CSS - Cascading Style Sheets | Aqua text |  |
| Vector img |  | Red text |  |
| Vector img |  | Green text |  |
| Text | * Grid system (Bootstrap) * Flexbox | Orange text | One by one after mentioning |
| 4 | Після освоєння всього вище перерахованого, настає час перейти до фреймворків. Такими можуть являтися bootstrap 4, який зараз в принципі найпопулярніший, або foundation, у якого також є свої користувачі. Кожен з цих фреймворків пропонує низку вже готових комплексних рішень найчастіше використовуваних елементів сайту. Таких як слайдери, форми, кнопки, меню і тд. Також в них є рішення по Grid system. Тобто, за допомогою цих фреймворків можна зробити сайт і без знань CSS та HTML. Вибрати з цих фреймворків я рекомендую один, особисто я використовую bootstrap 4, і то в основному тільки саму Grid system без дизайну, бо в будь якому випадку кожен сайт в комерційних цілях потрібно робити з новим дизайном під замовника.  Для пришвидшення розробки рекомендую розібратися в препроцесорах CSS - а саме SASS, SCSS, LESS. Особисто я віддаю перевагу препроцесору SASS. Препроцесор це код який ви пишете і він може трохи відрізнятися від CSS в плані синтаксису, але в результаті він всерівно буде компілюватися в CSS за допомогою спеціальних програм. В препроцесорах ще є такий неймовірний плюс як використання міксинів, які, наприклад, дають можливість замінити цілий участок коду однією змінною, яка при компіляції зміниться кодом CSS.  В кінці кінців, крім основних тегів, потрібно ще знати основи CSS3 і чим він відрізняється від CSS, також знати основи анімації, про-дивитися найпопулярніші бібліотеки, знати як працювати зі шрифтами, як зробити сайт адаптивним, і знати що таке очистка float, бо новачки на цьому дуже часто спотикаються. Для практики і, як на мене це просто цікаво, можете переглядати дизайни з сайту codepen.io, там дійсно бувають просто неймовірні рішення. | Text | * bootstrap 4 * foundation | Orange text | One by one after mentioning |
| Text | * Slider * Form * Menu * Button * Grid system | Red text | One by one after mentioning |
| Text | * Preprocessors:   + SASS,   + SCSS,   + LESS * Mixins | Green text | One by one after mentioning |
| Text | * CSS3 * найпопулярніші бібліотеки * шрифти * очистка float * codepen.io | Dark blue text | One by one after mentioning |
| 5 | Якщо ви вивчили все про що я казав вище, то на цьому єтапі ви вже готові для верстки сайтів з PSD. Можете підібрати в інтернеті декілька цікавих варіантів дизайну сайту в форматі PSD і спробувати їх перенести у код заради практики і закріплення основи. Цей крок є дуже важливим. Не хвилюйтеся, що ви щось не знаєте. На цьому етапі Google, StackOverflow, w3schools та css-tricks повинні бути вашими найкращими друзями. Кожного разу, коли ви щось не знаєте, просто шукайте в інтернеті, бажано англійською мовою. Тому одразу привчайтеся формувати пошукові запити саме нею.  Перфекціонізм це добре, але в помірній кількості. Не обов’язкото знати всі терміні на зубок, бо такий підхід може сповільнити процес навчання, тому дозвольте собі інколи помилятися та швидше отримувати більше знань. В наступному уроці ми продовжимо розглядати технології які потрібно знати frontend розробнику. | Vector img | Related image | Dark blue text |  |
| Text | * Google, * StackOverflow, * w3schools * css-tricks | Green text | One by one after mentioning |

**МІКРОЛЕКЦІЯ №3**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEXT  ITEM № | ТЕКСТ / ТРИВАЛІСТЬ ДЕМОНСТРАЦІЇ | ТИП | Що має бути показано | Коли має бути показано | Як має бути показано |
| 1 | Цей урок ми розпочнемо зі знайомства з JavaScript. Він є наступним кроком, після практики верстки сайту з PSD. При створенні передбачалося, що JavaScript буде використовуватися для простих речей. Ніхто не міг передбачити, що він буде застосовуватися у великих програмах і буде настільки поширеним.  Дуже часто люди плутаються і вважають що JavaScript і ECMAScript різні мови програмування. Але насправді ECMAScript - це просто більш технічне ім'ям для JavaScript. ES2015 / ES5 / ES6 - це скорочення від ECMAScript. ES2015 - це той самий ES6. Цифри в назві - це лише різні версії Javascript. На даний момент ES6 являється останньою версією мови Javascript.  JavaScript багато років розвивається. Є багато бібліотек, структур і фреймворків, які дозволяють писати код простіше, краще і швидше. Почнемо з бібліотек. Однією з найпопулярніший зараз являється jQuery побудована на основі JavaScript, яка використовується для спрощення та прискорення розробки. В jQuery використовується інший синтаксис, наприклад для роботи з DOM або Document Object Model, що являє собою інтерфейс, який розглядає HTML документ як структуру, де кожен тег є об'єктом, що представляє частину документа. Тому, я спочатку рекомендую вивчити справжній JavaScript, бо його в будь якому випадку його потрібно розуміти і вміти використовувати, коли це потрібно. Після цього вже можна починати користуватися jQuery для пришвидшення розробки, адже він, як на мене, зручніший ніж JavaScript. Також, дуже популярними бібліотеками, які я рекомендував би вивчати після jQuery будуть: Owl Carousel, Magnificent pop-up та Slick. Ці бібліотеки я використовую на постійній основі. | Text + Vector img | JavaScript Image result for JavaScript | Green text |  |
| Text | ECMAScript = JavaScript | Red text |  |
| Text | * ES2015 / ES5 / ES6 = ECMAScript * ES2015 = ES6 * ES6 | Dark blue text | One by one after mentioning |
| Text | jQuery | Orange text |  |
| Text | DOM - Document Object Model | Purple text |  |
| Text | * Owl Carousel * Magnificent pop-up * Slick | Aqua text | One by one after mentioning |
| 2 | Якщо ви збираєтеся дійсно глибоко почати розбиратися в frontend розробці, а особливо в побудові SAP або Single Page Application, ви, ймовірно, захочете вивчити фреймворки JavaScript. Першим з них буде React.js - дуже популярний фреймворк, особливо для стартапів, дуже швидко працює, дуже легкий, він використовує менеджера під назвою Redux. Технічно React.js це бібліотека, але ви можете легко додати до нього матеріали, щоб зробити його повноцінним фреймворком.  Наступним є Angular, який теж дуже популярний, особливо на підприємствах та в бізнесі. На відміну від React, з коробки в Angular більше функцій, і ,можна сказати, що все включено для того щоб вважати його повноцінним фреймворком,.... Angular має менеджера під назвою NgRx, я з ним не працював, але з того, що я знаю, він дуже схожий на Redux.  Останнім з найпопулярніших фреймворків є Vue.js - я з ним не працював, але кажуть, що йому найпростіше навчитися, має хороший синтаксис, і в нього також є менеджер під назвою vueX. | Text | SAP - Single Page Application | Red text |  |
| Text  +  Vector img | * React.js   + Redux * Angular   + NgRx * Vue.js   + vueX | Orange words | One by one after mentioning |
| 3 | Тепер поговоримо про інструменти, які допоможуть вам у створенні, збірці та розробці додатків на JavaScript. Ця категорія включає linters, task runners та bundlers.  Task runners або виконувачі завдань допомагають пришвидшити розробку автоматизуючи деякі процеси. Одними з найпопулярніших task runners зараз є gulp і grunt. Загальні завдання включають в себе такі речі, як перегляд змін файлів, об'єднання, мінімізація та префіксація файлів. Ці завдання часто виконуються за допомогою консолі.  linters - це засоби, які аналізують вихідний код, щоб відзначити помилки та підозрілі конструкції. Існує багато варіантів таких програм, включаючи JSLint, JSHint та JSCS. Але в даний час найпопулярнішою являється ESLint.  Бландери, такі як Parcel, Webpack, Rollup керують і завантажують взаємозалежні модулі. Таким чином вони вступають в роботу під час збірки вашого сайту, bundler підключає всі модулі разом в один файл (пакет). Для початку знайомства виберіть один бландер, бажано щоб це був Webpack, бо на даний момент він найпопулярніший.  Знання Javascript відкривають величезні можливості розробки. Якщо ви його знаєте, ви можете створювати мультиплатформенні додатки. Стаючи frontend девелопером, важливо знати JavaScript, але також пам'ятайте, що для того щоб ним по справжньому оволодіти, потрібно знати базу, а саме: HTML та CSS. JS є великим плюсом, але на першому етапі буде достатньо просто знати його синтаксис, бібліотеку jQuery та отримати досвід написання базових програм, наприклад: модальні вікна, слайдери, акордеон. | Text | * linters * task runners * bundlers | Red text | One by one after mentioning |
| Text | * task runners:   + gulp   + grunt | Green text | One by one after mentioning |
| Text | * linters   + JSLint   + JSHint   + JSCS   + ESLint | Dark blue text | One by one after mentioning |
| Text | * Bundlers   + Parcel   + Webpack   + Rollup | Orange text | One by one after mentioning |
| Text | * Синтаксис * Бібліотека jQuery * Базові програми:   + Модальні вікна   + Слайдер   + Акордеон | Aqua text | One by one after mentioning |
| 4 | Що стосується додаткових програм, то тут все доволі просто. Потрібно пройтися по основним пунктам, а саме: редактори, браузери, редакція картинок, програма для зв'язку з сервером та програма керування версіями сайту.  Для початку виберіть зручний для вас редактор коду. Це може бути Sublime Text 3 і я рекомендую саме цей редактор, але також непоганими являються Atom, VSCode та Brackets. Так як наш сайт повинен бути кроссбраузерным, користуйтеся Chrome та Firefox. При бажанні, можете ще якимись. Як редактор картинок використовуйте Adobe Photoshop, а для векторів можете вибрати Inkscape або Adobe Illustrator. Для зв’язку з хостом зараз найпопулярнішим FTP клієнтом являється FileZilla. Якщо ви хочете керувати віддаленим сервером через протокол SSH, то використовуйте Putty.  Остання допоміжна програма це GitHub, який являється веб-службою для керування версіями проекту та вихідним кодом. Тобто якщо ви зробили якусь серйозну помилку, то можна перейти на попередню версію сайту, також цей сервіс являється дуже зручним для розробки великих проектів з великою кількістю програмістів. На додачу, у GitHub є дуже корисна функція під назвою Gist. Це маленькі уривки коду, які ми зберігаємо на сервісі GitHub і в будь-який момент можемо отримати до них достум прямо з редактора, що дуже допомагає при розробці. | Text | * Редактор * Браузер * Редакція картинок * Програма для зв'язку з сервером * Програма керування версіями сайту | Aqua text | One by one after mentioning |
| Text  +  Vector img | * Sublime Text 3 * Atom Image result for atom text editor * VSCode * Brackets Image result for Brackets text editor | Orange text |  |
| Text  +  Vector img | * Chrome * Image result for Adobe PhotoshopFirefox | Blue text |  |
| Text  +  Vector img | * Adobe Photoshop * Inkscape or * Related imageAdobe Illustrator | Tan text |  |
| Text  +  Vector img | * FileZilla Image result for FileZilla * SSH -> Putty | Red text |  |
| Text  +  Vector img | * Image result for githubGitHub   + Gist | Green text |  |
| 5 | З книжок по frontend я можу порекомендувати дві книжки авторства Jon Duckett. А саме HTML & CSS: Design and Build Web Sites і JavaScript and JQuery: Interactive Front-End Web Development. Книги 2011 і 2013 року випуску. Я вибрав ці книги бо вони стали бестселлерами і в них дуже гарні ілюстрації, тобто якість на дуже високому рівні. Я вивчав їх англійською мовою, але в них наче є переклад. В будь якому випадку, книг і матеріалів існує безліч і ви зможете знайти те, що вам подобається. Пам'ятайте, що навчання - це процес, і просто читання не допоможе вам багато чого досягти, тому практикуйтеся.  Веб-сайти, які все ще допомагають мені в моїй повсякденній роботі, - це w3schools і мережа розробників Mozilla (MDN). Я також рекомендую вам використовувати саме їх, але зауважте, що вся документація там англійською мовою. Багато хто каже, що w3schools не дуже гарне джерело інформації, але я не з цим не погоджуюсь. Коли я працюю, і щось забув, але в принципі добре знаю тему, то документація на цьому сайті мені дуже сильно допомагає, адже являється набагато чіткішою і коротшою, ніж та, яка розміщення на MDN.  Я б сказав, що w3schools є найкращим ресурсом для відновлення інформації, а MDN - це найдосконаліша документація, що ведеться, і підтримується відомими компаніями, і використовувати її потрібно для розуміння складних концептів.  А на цьому ми закінчуємо наш урок про технології frontend. З вами був Олексій Федоренко, а в наступному відео ми розглянемо що ж потрібно знати backend програмісту. | Vector img |  | Green text |  |
| Text | * w3schools * Mozilla | Orange text |  |

**МІКРОЛЕКЦІЯ №4**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEXT  ITEM № | ТЕКСТ / ТРИВАЛІСТЬ ДЕМОНСТРАЦІЇ | ТИП | Що має бути показано | Коли має бути показано | Як має бути показано |
| 1 | В цьому відео я розкажу про мови програмування саме для web backend, які на мою думку можна вивчати в 2018. Також дам поради на рахунок того де ці мови використовуються і що краще вивчати новачку. Що стосується самої професії backend розробник то тут ви будете писати логіку веб-сайтів і якщо порівнювати сайт з машиною, то HTML - каркас, css - дизайн, javaScript - вся електроніка, а backend - це двигун і всі механізми, які працюють на паливі, змушуючи нашу машину зрушити з місця. Якщо ж переводити цю простеньку алегорію в світ вебу, то backend програмування - це вся логіка веб-сайту. Саме завдяки backend мовам у вас є можливість зареєструватися на сайті, переписуватися в месенджерах і так далі - це все логіка, це все backend програмування. Ось тільки замість палива мови backend споживають ресурси комп'ютера, а саме процесор, ОЗУ і місце на жорсткому диску.  Ну а перед тим як взагалі задуматися про те щоб вивчати серверну розробку я рекомендую вам подивитись відео про frontend і хоча б на поверхневому рівні його вивчити. В плані математики backend програмування є складнішим по відношенню до frontend, так як ви створюєте логіку сайту, яка може включати в себе дуже складні математичні концепції. Також важливо щоб ви знали як працює інтернет і як з ним взаємодіяти. А саме, щоб вас не лякали хоча б такі слова як HTTP, websocket і як відбувається передача даних. Більш того, ви вже повинні вміти вільно працювати з комп'ютером, розуміти як він влаштований, хоча б на мінімальному рівні розуміти його hardware.  Звичайно, ви можете почати вивчення серверної розробки і без знання всього наведеного вище але в цьому випадку не гарантовано те, що ви будете розуміти як працює сервер, а значить буде й нерозуміння того працює ваш код і як його писати правильно, щоб ефективно вирішувати поставлені завдання.  Існують мови програмування backend які одразу писалися для вебу і працювати на них дуже зручно і швидко. Це такі мови як PHP і JavaScript. У цих мов дуже сильне ком'юніті і багато документації. Є мови більш загальної направленості, але також дуже популярні для backend розробки. Такими мовами являються Python, Ruby, Java і C++. Тепер по кожній більш детально. | Vector img |  | Orange text |  |
| Vector img |  | Red text |  |
| Vector img |  | Dark blue text |  |
| Vector img |  | Green text | Overlay approved over math |
| Text | * HTTP * websocket * hardware | Aqua text |  |
| Text | * PHP * JavaScript * Python * Ruby * Java * C++ | Tan text |  |
| 2 | JavaScript є мовою для frontend програмування, але існує середовище виконання під назвою Node.js, яке виконує код JavaScript на стороні сервера. Впринципі Node.js це звичайний  JavaScript, але розширений і призначені для роботи на сервері, іншими словами це серверний JavaScript. Слідуючи з вищесказаного, вам спочатку потрібно вивчити саму мову JavaScript, а вже потім вивчити особливості Node.js.  В останній час платформа Node.js стала особливо набирати популярність. Перевагою є те що по факту JavaScript, працює на обох сторонах - сервер та клієнт, або frontend і backend. Цього може бути достатньо для того, щоб закохатися в Node.js, але це ще не все. Node.js використовує не блокуючу модель вводу-виводу, що робить її легкою та ефективною. Це означає, що він може бути використаний для створення швидких та масштабних real-time мережевих програм, які отримають тисячі одночасних запитів. Прикладом роботи такої системи являються чати, де ми майже миттєво отримуємо повідомлення від іншого користувача. Ці переваги зараз дуже часто починають використовувати стартапери при розробці своїх проектів.  Ще однією причиною для вивчення Node.js є Express та npm. Обидва ці інструменти зосереджуються на таких речах, як повторне використання коду та вартість розробки. Express, є основою для веб-додатків на Node.js. Вона призначена для побудови веб-програм і інтерфейсів API. Він розширює оригінальний набір програмного забезпечення (SDK) з набором функцій для розробки веб та мобільних додатків.  Npm, що є скороченням від (Node Package Manager) це перш за все це онлайн-сховище для публікації проектів на Node.js з відкритим кодом. По-друге, це утиліта командного рядка для доступу до проектів завантажених в це сховище. Ви можете навіть повторно використовувати код, написаний іншими розробниками. Щоб дати вам уявлення про розмір цього сховища, npm містить понад 200 000 пакетів.  Підсумовуючи по Node.js, його можна вивчати навіть прямо після основ frontend та того як працює інтернет, але поріг входу на ринок, вище ніж в PHP. | Vector img |  | Tan text |  |
| Vector img |  | Red text |  |
| Text | Real-time processing | Dark blue text |  |
| Text | * Express   + API   + SDK * Npm - Node Package Manager | Green text |  |
| 3 | Хоч і кажуть вже років 10 під ряд що PHP вимирає, що це дуже непродума мова програмування і тд, але я з цим не згоден. Якщо подивитися на статистику по зростанню популярності, то у PHP вона залишається майже сталою вже на протязі багатьох років. Тому, щоб про PHP не казали, але в кінці-кінців повноцінної заміни йому сьогодні як такої немає. Ні Python, ні Ruby навіть JavaScript не є повноцінною заміною для PHP, так як є мовами більш загально спрямованими, тобто не створеними конкретно для web backend. В PHP є своя ніша. На даний момент більшість сайтів в інтернеті працює саме на ньому.  Для зв’язку з сервером PHP потрібно використовувати щось по типу моста, наприклад технологію AJAX. Саме завдяки цій технології ми можемо серйозно скоротити час завантаження інформації, закачуючи з сервера не цілу сторінку, а якусь частину, яку ми згодом позиціонуємо завдяки JavaScript. Саме цей процес лежить в основі динамічних сайтів.  При цьому майже всі системи управління контентом або CMS працюють саме на PHP. Наявність таких систем, як готових рішень дуже пришвидшує роботу і дає можливість створити сайт навіть без знання мови програмування, використовуючи готові шаблони. Саме тому, майже всі інтернет магазини працюють на PHP. Прикладом CMS написаної на PHP є WordPress, яка також є найпопулярнішою CMS у світі. Також популярними CMS зараз на ринку являються Magento, Joomla, Drupal, OpenCart, PrestaShop. Серед них OpenCart і PrestaShop спеціалізуються саме на інтернет магазинах.  На PHP є дуже багато фреймворків, основними з яких являються Laravel і Codelnighter. Найпопулярнішим є Laravel, виразною особливістю якого є елегантний синтаксис. Основна задача фреймворків - це полегши загальні завдання, що використовуються в більшості веб-проектів, таких як аутентифікація, сесії та кешування. | Vector img |  | Green text |  |
| Vector img |  | Red text |  |
| Vector img |  | Dark blue text |  |
| Vector img |  | Orange text |  |
| Text | * WordPress * Magento * Modx * Joomla * Drupal * OpenCart * PrestaShop | Aqua text |  |
| Text  + Vector img | * Laravel * CodeInighter |  |  |
| 4 | Ще одним плюсом PHP є те, що він дуже зручний і на ньому можна дуже швидко розробляти сайти, він доволі простий і дозволяє замовнику серйозно економити на розробці. Більше того, бажаючому вивчити backend розробку, PHP готовий надати відмінні можливості і вкрай низький поріг входження в сферу. Тобто, наприклад, щоб написати свій перший блог з нуля, можна в цілому витратити близько двох-трьох тижнів на вивчення цієї мови в порівнянні з тим же Python де вам спочатку потрібно вивчити мову, а тільки потім вже який-небудь веб-фреймворк, наприклад django, щоб написати мінімальний сайт. Тим більше PHP розвивається і стає з кожною версією кращим і швидшим. Наприклад при переході з версії 5.х на версію 7.х швидкість роботи виростає в два рази. Нинішня актуальна версія PHP це 7.х, але половина сайтів досі працює на 5.х версіях.  При врахуванні всіх переваг PHP, а особливо фактор ціна-якість для замовника, я думаю що PHP навряд чи вимре в найближчому майбутньому. Тому, якщо ви новачок і плануєте якнайшвидше вивчити backend і почати працювати девелопером, то PHP ідеальний для вас вибір. Але потрібно зауважити, що у випадку PHP найбільше ціняться саме Full-Stack девелопери, тому перед вивченням PHP я рекомендую вивчити спочатку frontend на рівні близько мідла.  В наступному уроці ми продовжимо розглядати технології які потрібно знати backend розробнику. | Vector img |  | Aqua text |  |
| Vector img | to | Green text |  |
| Vector img |  | Red text | Cross out the img |

**МІКРОЛЕКЦІЯ №5**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEXT  ITEM № | ТЕКСТ / ТРИВАЛІСТЬ ДЕМОНСТРАЦІЇ | ТИП | Що має бути показано | Коли має бути показано | Як має бути показано |
| 1 | Цей урок ми розпочнемо зі знайомства з мовами програмування Python, Ruby, Java і C++. В цих мовах я не особливий експерт, бо на них не працював, адже неможливо знати всі мови зразу. Але я в свій час їх вчив і маю декілька слів по кожній.  Веб-розробка на Python сьогодні має доволі сильні позиції, особливого зростання популярності в неї не має, але можливості які відкриваються при використанні Python доволі вражаючі. На цій мові написано основна частина функціоналу youtube а також деякі частини в яндекса і Google крім цього на ньому працюють багато відомих сервісів. Веб-сайти на Python розробляють за допомогою фреймворків, найвідомішим з яких є django. Якщо у вас немає якихось специфічних цілей, то бажано вивчати саме цей фреймворк через очевидні плюси всіх популярних систем, а саме: в нього є велика підтримка ком'юніті і розробників.  Я би рекомендував вивчати Python для вебу тільки у тому випадку коли ви вже гарно знаєте одну з основних мов для web backend, а саме PHP чи JavaScript, або ви вже дуже добре знаєте Python і хочете спробувати себе в вебі. | Text  + Vector img | * Python * Ruby * Java * C++ | Red text |  |
| Vector img |  | Dark blue text |  |
| 2 | Також в 2018 році залишається тренд на Ruby. Використовують його завичай за швидкість роботи, або дуже підходящий функціонал для проекту. Для написання бекенду сайту на мові Ruby використовують фреймворк під назвою Ruby on Rails.  На рахунок Java, C++. Ці мови являються дуже загальнонаправленими і вивчати їх новачку чисто для того щоб програмувати backend сайту неправильно бо ці мови часто використовують тільки для певних специфічних цілей, де важливий професіоналізм. Тому навчання може сильно затягнутися, бо спочатку потрібно вивчити мову, а потім вже фреймворк. Java більше використовується для Mobile Dev, а у випадку C++ це Game Dev.  В принципі, рекомендації по вивченню залишаються такими самими як і для Python. Їх можна вивчати як додаткову мову, або якщо ви вже добре знаєте цю мову і хочете спробувати себе у вебі.  Саме важливе що ви повинні розуміли при виборі мови програмування - це те, що найкраща мова програмування, та, яку ви знаєте найкраще. В будь якому випадку, якщо ви будете професіоналом, то вашим навичкам можна буде знайти застосування і вам навіть не будуть заважати Індуси, які віддаються за копійки, бо й якість в них відповідна. Я би рекомендував використовувати одну мову як основну, другою займатися як хобі. І просто розуміти синтаксис інших.  Якщо ви новачок в програмуванні, то рекомендую все-таки звести свій вибір саме до PHP та Node.js. Адже вони найбільш широко застосовуються для вебу. Також вибираючи основну з цих мов, було б непогано хоча б на початковому рівні знати іншу, щоб коли завгодно можна було змінити профіль. | Text  + Vector img | Ruby on Rails | Dark blue text |  |
| Vector img |  | Green text |  |
| Text | * Java -> Mobile Dev * C++ -> Game Dev | Purple text |  |
| Text  + Vector img | + | Orange text | Show one by one and then overlay approved over the languages |
| 3 | Зараз ми поговоримо про DMBS (DataBase Management System) або українською це буде СУБД (Система Управління Базами Даних). Вони являються невід'ємною частиною програмування на бекенд. В них зберігається вся інформація на вивід якої роблять запити мови бекенд. Якщо послухати розробників, які говорять про бази даних, зазвичай можна почути такі слова, як надійні, масштабовані, ефективні та ін. Обговорення зосереджуються на силі та її інтеграції з іншими технологіями.  До того як почати вивчати СУБД, я рекомендую розібратися як працює SQL. Вивчити як створювати складні запити між таблицями. Також дуже важливо зрозуміти як працюють самі бази.  Вивчаючи backend мови програмування ви побачите, що ваше життя буде набагато простішим, якщо ви виберете правильну СУБД, створену для роботи з вашою веб-технологією. Наприклад, розробники PHP зазвичай використовують MySQL або Oracle, для Java краще підійде DB2, а для Node.js краще за все підійде СУБД MongoDB.  Новачку, я все таки б рекомендував вибрати MySQL у зв’язці з PHP, оскільки вона має хороші інтерфейси користувача та чудову документацію в Інтернеті. | Text | DMBS - DataBase Management System | Orange text |  |
| Vector img |  | Purple text | Change one by one |
| Vector img |  | Red text |  |
| Text | * PHP -> MySQL/Oracle * Java - DB2 * Node.js - MongoDB. | Green text | Turn up one by one |
| 4 | Також нагадую, що мову ідеально вивчити не вийде, і до кожного нового проекта вам скоріше за все доведеться шукати нову інформацію в інтернеті. Навчання ніколи не завершиться і якщо ви хочете бути програмістом, доведеться з цим змиритися. Максимум чого можна досягти це рівня знань на якому ви по праву будете вважати себе новачком, середняком або експертом. Більш того, мови програмування постійно поліпшуються і розширюються, а технології не стоять на місці. Цим все і пояснюється.  З книжок по backend я можу порекомендувати Робін Ніксон - Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS і HTML5. Я вивчав цю книгу бо там є пояснення як працює веб, як працювати з базами, що таке PHP, основи динамічного вебу, але саме головне, що там були практичні завдання. Для вивчення Node.js я використовував офіційну документацію і мені було в принципі достатньо, адже я вже знав JavaScript. Ще раз нагадаю будь якому випадку, книг і матеріалів існує безліч і ви зможете знайти те, що вам подобається. | Vector img |  | Green text | Cross out the img |
| Text | * Junior * Middle * Senior | Dark blue text |  |
| Vector img |  | Orange text |  |
| Text | Node.js – офіційна документація | Aqua text |  |
| 5 | Читати теорію це звичайно дуже важливо, але найголовніше це практика. Я би навіть сказав 20% теорія, 80 - практика.  Тому ще раз подумайте навіщо вам знати web backend,  бо щоб стати непоганим спеціалістом майже з нуля, вам доведеться витратити рік або два на повному робочому дні, або близько 2000 годин. Тобто це вже той рівень, на якому ви вже можете вийти на непогану зарплату. Хоча ці цифри можуть дуже сильно відрізнятися від людини до людини. Я кажу судячи по собі і трошки включаючи загальну статистику. Хоча в мене ці 2000 годин розтягнулися аж десь на 4 роки через те що я поєднував вивчення програмування з офіційним навчанням.  А на цьому все. З вами був Олексій Федоренко, удачі вам у становленні веб розробником. | Text | 20% - теорія  80% - практика | Aqua text |  |