

ابدأ البرمجة من الصفر

كتاب الكورس — من سؤال «يعني إيه برمجة؟» لأول مشروع تنزله

إعداد: محمد ناصر

وفي كل فصل مكان لبلاي ليست فيديوهات

MOONASSER

محتويات الكتاب

الجزء الأول — من الصفر خالص — يعني إيه برمجة؟

- الفصل 1: يعني إيه برمجة؟
- الفصل 2: الكمبيوتر يفهم كلامك إزاي؟
- الفصل 3: كلمات هتسمعها كتير
- الفصل 4: نجهز الجهاز
- الفصل 5: إزاي تذاكر صح؟

الجزء الثاني — الويب: HTML و CSS

- الفصل 6: HTML — عضم الصفحة
- الفصل 7: الفورمات و Semantic HTML
- الفصل 8: CSS — الشكل والألوان
- الفصل 9: Grid و Flexbox
- الفصل 10: الصفحة على الموبايل
- الفصل 11: مشروع ١ — صفحتك الشخصية

الجزء الثالث — JavaScript والحركة في الصفحة

- الفصل 12: JavaScript — الأساسيات
- الفصل 13: DOM — تعديل الصفحة بالكود
- الفصل 14: مشروع ٢ — تطبيق مهام

الجزء الرابع — المسارات: تختار إيه؟

- الفصل 15: تختار مسارك إزاي؟
- الفصل 16: Frontend
- الفصل 17: Backend
- الفصل 18: Full-Stack
- الفصل 19: الموبايل
- الفصل 20: الألعاب
- الفصل 21: الروبوتات
- الفصل 22: الأمن السيبراني
- الفصل 23: الذكاء الاصطناعي

الجزء الخامس — الأدوات والـ AI وخطة مذاكرة

- الفصل 24: Git و GitHub
- الفصل 25: أدوات الـ AI للمبرمج
- الفصل 26: خطة ٣ شهور واقعية

الجزء الأول

من الصفر خالص — يعني إيه برمجة؟

MOONASSER

الفصل 1 — يعني إيه برمجة؟

قبل أي كود، خلىنا نفهم بنعمل إيه أصلاً

الحكاية ببساطة

البرمجة إنك تقول للكمبيوتر يعمل إيه، خطوة خطوة. وهو مش هيزود ولا هينقص ولا هيفهم قصداً، هو بيعمل اللي انت كتبتة بالحرف. مش ذكي يعني، بس سريع جداً وينفذ من غير زهق.

لو قلت لصاحبك «هات لي مية» هو فاهم. الكمبيوتر لأ. لازم تقوله: قوم، امشي للمطبخ، افتح الدولاب، هات كوباية، افتح الحنفية، املاها، اقل الحنفية، ارجع. الطريقة دي في التفكير هي اللي هتشتغل بيها في كل حاجة بعد كده.

- البرمجة = تكسر المشكلة الكبيرة لخطوات صغيرة تافهة.
- اللغة مش هي الأساس، الأساس هو دماغك وطريقة تفكيرك.
- مافيش حاجة اسمها «مش هينفع معايا» — الموضوع تكرر ووقت وخلاص.

طيب البرمجة بتعمل إيه؟

- مواقع وتطبيقات على النت (websites & web apps)
- تطبيقات موبايل (mobile apps)
- ألعاب (games) ببرامج زي Unity
- روبوتات وأجهزة (robotics)
- ذكاء اصطناعي وشغل بيانات (AI & data)

نفكر زي المبرمج

قبل ما تكتب كود، اكتب المطلوب في جملة واحدة. بعدها قسمه لمدخلات، خطوات، ونتيجة. لو مش عارف تشرح الحل على ورقة، غالباً الكود هيطلع ملخبط.

مثال: برنامج يحسب متوسط ٣ درجات. المدخلات هي الدرجات، والخطوات جمعهم وقسمتهم على ٣، والنتيجة رقم المتوسط. الكود هنا مجرد ترجمة للخطة دي.

- ابدأ بالمشكلة، مش باسم اللغة.
- جرب الحل على مثال صغير بإيدك قبل ما تكتبه.
- كل خطوة في الخطة لازم تكون واضحة ومحددة.

مثال: خطوات بالترتيب — دي فكرة البرمجة كلها

```
STEP 1 -> open the cupboard
STEP 2 -> take a glass
STEP 3 -> fill it with water
STEP 4 -> give it to your friend
```

تمرين الفصل

- اكتب خطوات عمل كوابية شاي بالتفصيل الممل، ووريها لحد يقرأها بالحرف.
- اكتب ٣ حاجات نفسك تعملها بالبرمجة.

بلاي ليست الفصل: يعني إيه برمجة

الرابط: _____

الفصل 2 — الكمبيوتر يفهم كلامك إزاي؟

من اللي انت كاتبه لحاجة شغالة فعلا

من الكود للتنفيذ

الكمبيوتر في الآخر مش شايف غير 0 و 1. الكلام اللي بتكتبه بيتحول بحاجة تانية: يا إما compiler يترجم الكود كله مرة واحدة قبل ما يشتغل (زي ++C)، يا إما interpreter يمشي سطر ورا سطر وهو شغال (زي Python و JavaScript). مش لازم تحفظ الفرق دلوقتي. المهم تعرف إنك لو نسيت فاصلة أو قوس، الكمبيوتر هيقف ويقولك في غلط — وده مش هيزعلك بعد شهر، هيبقى عادي.

- Compiled: يشتغل أسرع ويمسك أغلاط كتير قبل التشغيل
- Interpreted: أسهل في البداية ويتجرب بسرعة
- المتصفح جواه JavaScript ل interpreter، فانت مش محتاج تنزل حاجة تبدأ.

رحلة السطر اللي كتبته

الكود بيتقري حسب قواعد اللغة. لو القواعد سليمة يبدأ التنفيذ، ولو فيه غلطة تركيبية هتشوف Syntax Error. أما لو الكود اشتغل وطلع نتيجة غلط، دي Logic Error ودي محتاجة تراجع تفكيرك.

رسالة الخطأ مش عقاب. اقرا أول سطر مفهوم فيها، بص على اسم الملف ورقم السطر، وبعدها راجع آخر حاجة غيرتها.

- Syntax Error: طريقة الكتابة غلط
- Runtime Error: المشكلة ظهرت وقت التشغيل
- Logic Error: البرنامج اشتغل بس النتيجة غلط

مثال: نفس الفكرة بلغتين مختلفتين

```
print("hello")          # Python
console.log("hello");    // JavaScript
```

تمرين الفصل

- افتح Console في المتصفح واكتب `console.log(2 + 2)`.
- دور على معنى compiler و interpreter بكلامك انت في سطرين.

بلاي ليست الفصل: إزاي الكمبيوتر يفهم الكود

الرابط: _____

الفصل 3 — كلمات هتسمعها كثير

قاموس صغير بيوفر عليك لخبطة شهو

الأساسيات

- علة بتحت فيها قيمة: اسم، رقم، أي حاجة — Variable.
- (boolean) صح/غلط، (number) رقم، (string) نوع القيمة: نص — Data type.
- (if / else) لو كذا اعمل كذا، وغير كده اعمل كذا — Condition.
- (for / while) كرر الخطوة دي كذا مرة — Loop.
- شنة خطوات ليها اسم، بتناديها وقت ما تحتاجها — Function.
- لسة قيم بترتيب — Array.
- حاجة فيها خصائص، كل خاصية ليها اسم وقيمة — Object.
- إنك تدور عليها — Debugging. حاجة بتحصل غلط في البرنامج — Bug.
- الخطة اللي هتحل بيها المشكلة — Algorithm.
- كود جاهز عمله ناس قبلك بيوفر عليك وقت — Library / Framework.
- الطريقة اللي برنامج بيكلم بيها برنامج ثاني — API.
- Git فولدر مشروعك وهو متسجل على — Repo.

المصطلحات وهي شغالة

- المتغير بيحفظ قيمة، والشرط بياخد قرار، وال loop يكرر، وال function تجمع خطوات ليها اسم. لما تركيب الأربع حاجات دول مع بعض تقدر تعمل برامج كثير جدا.
- متقلقش من الإنجليزي. اكتب عندك قاموس صغير: المصطلح، معناه بكلامك، ومثال من سطر واحد. بعد كام مشروع هتلاقي الكلمات ثبتت لوحدها.
- احفظ الفكرة بالمثال، مش التعريف.
 - نفس المفهوم موجود في أغلب لغات البرمجة.
 - ارجع للقاموس وقت ما تنسى، النسيان طبيعي.

مثال: المصطلحات دي في ٨ سطور كود

```
let name = "Ahmed";      // variable
let age = 17;             // number
let ok = true;            // boolean
let skills = ["html","css"]; // array
let user = { name, age }; // object
if (age > 15) { }         // condition
for (let i = 0; i < 3; i++) { } // loop
function hi(n) { return "hi " + n; } // function
```

تمرين الفصل

- اكتب كل مصطلح ومعناه بكلامك — بلاش نسخ.
- حدد المصطلحات اللي لسه مش واضحة وارجلها بعد أسبوع.

بلاي ليست الفصل: المصطلحات الأساسية

الرابط: _____

الفصل 4 — نجهز الجهاز

حاجات بسيطة قبل أول سطر

اللي هتنزله

- متصفح: Chrome أو Firefox، وبتفتح الـ DevTools بـ F12.
- محرر كود: Visual Studio Code — مجاني وهو اللي كله بيشتغل عليه.
- إضافات مفيدة في VS Code: Live Server و Prettier و Auto Rename Tag.
- GitHub وحساب على Git.
- سيبها لبعدين لما تحتاجها — Node.js.

عادات ابدأ بيها من أول يوم

- فولدر لكل مشروع، وأسماء إنجليزي من غير مسافات.
- اسيف كتير (Ctrl+S) وبص على النتيجة في المتصفح على طول.
- اكتب بإيدك. Copy/paste وانت بتتعلم هو نفسه إنك مش بتتعلم.

ترتيب مشروعك من البداية

اعمل فولدر رئيسي للكورس، وجواه فولدر لكل مشروع. سمي الملفات بحروف إنجليزي صغيرة، واستخدم الشرطة بدل المسافة. الترتيب ده هيوثر عليك مشاكل كتير بعدين.

في VS Code افتح فولدر المشروع نفسه، مش ملف واحد. كده الإضافات وال terminal والبحث هيشغلوا على المشروع كله.

- هو أول ملف بيدور عليه أغلب الاستضافات index.html.
- متغيرش أكثر من حاجة قبل ما تشوف النتيجة.
- لو الصفحة مش بتتحديث، اعمل حفظ الأول.

مثال: أول ملف index.html تكتبه

```
<!DOCTYPE html>
<html lang="ar" dir="rtl">
  <head>
    <meta charset="utf-8" />
    <title>■■■■ ■■■■</title>
  </head>
  <body>
    <h1>■■■■■</h1>
  </body>
</html>
```

تمرين الفصل

- نزل VS Code و Live Server واعمل ملف index.html فيه اسمك.
- افتح الصفحة في المتصفح وغير العنوان وشوف التحديث فوري.

بلاي ليست الفصل: تجهيز الجهاز والأدوات

الرابط: _____

الفصل 5 – إزاي تذاكر صح؟

الطريقة أهم من إنك تلاقي أحسن كورس

حاجات بسيطة بس بتفرق

أغلب الناس مش بتسيب البرمجة لأنها صعبة، بتسيبها لأنها بتتفرج كثير وبتكتب قليل. لو اتفرجت على فيديو، اقفله واعمل اللي فيه بإيدك قبل تروح اللي بعده.

- ربع الوقت مشاهدة، والباقي كتابة كود.
- شغل بعد كل خطوة صغيرة، متستناش لآخر الفيديو.
- الأغلاط شغل عادي جداً. اقرا الرسالة بهدوء، هي بتقولك المشكلة فين.
- مشروع صغير خلصته أحسن من ١٠ كورسات ما خلصتهمش.
- كل أسبوع اسأل نفسك: اتعلمت إيه؟ ولسه إيه مش واضح؟

جلسة مذاكرة عملية

قسم الساعة: ١٠ دقائق مراجعة، ٢٠ دقيقة شرح، ٢٥ دقيقة تطبيق من غير الفيديو، ٥ دقائق تكتب اللي وقف معاك. بالطريقة دي كل جلسة ليها نتيجة واضحة.

لو وقفت في مشكلة، جرب ٢٠ دقيقة بنفسك: اقرا الخطأ، اطبع القيم، وابحث بالكلمات المهمة. بعدها اسأل، بس ابعث الكود ورسالة الخطأ وإيه اللي جربته.

- متجمعش مصادر كتير لنفس الموضوع.
- اعمل مشروع صغير بعد كل مجموعة دروس.
- راجع القديم أثناء بناء الجديد.

تمرين الفصل

- اعمل جدول أسبوع فيه ساعة في اليوم واكتب فيه اتعلمت إيه.
- بعد أي فيديو، اقفله واكتب اللي فيه من دماغك.

بلاي ليست الفصل: طريقة التعلم والمذاكرة

الرابط: _____

الجزء الثاني

الويب: HTML و CSS

MOONASSER

الفصل 6 — HTML — عضم الصفحة

أول حاجة هتكتبها بإيدك

إيه هو HTML؟

مش لغة برمجة، هي طريقة بتوصف بيها المحتوى: ده عنوان، ودي فقرة، ودي صورة، وده لينك. اعتبرها العضم اللي HTML الصفحة واقفة عليه، ولسه مافيش شكل ولا ألوان.

- العنصر بيبدأ ب tag ويقفل ب tag، وبينهم المحتوى.
- ال Attributes معلومات زيادة للعنصر: href, src, alt, class, id.
- كل صفحة بتبدأ ب <DOCTYPE html> وجواها head و body.

حاجات هتستخدمها كل يوم

- العناوين: من h1 ل h6 — و h1 واحد بس في الصفحة.
- النصوص: p, strong, em, br, hr.
- الروابط والصور: a, img — ومتناساش alt للصورة.
- القوائم: ul, ol, li.
- التقسيم: div, span, section, header, footer, nav, main.
- الجداول: table, tr, th, td.

اقرا الصفحة كشجرة

- كل عنصر ممكن يبقى جواه عناصر ثانية. ال body هو الأصل، وجواه header و main و footer، وكل قسم جواه عنوان ومحتوى. الترتيب الصح بيخلي الصفحة أسهل في التنسيق والقراءة.
- ال class اسم تقدر تكرره على عناصر كتير، لكن ال id المفروض يكون لعنصر واحد. استخدم class للشكل، و id لما تحتاج توصل لعنصر محدد.
- اقفل كل tag فتحته.
- متستخدمش br علشان تعمل مسافات؛ دي شغلة CSS.
- اكتب alt يشرح الصورة اللي مش شايفها.

مثال: عناصر بتستخدمها كل يوم

```
<h1>■ ■ ■ ■ ■ </h1>
<p>■ ■ ■ ■ ■ ■ ■ ■ ■ ■ <strong>■ ■ ■ ■ ■ </strong>.</p>
<a href="https://moonasser.site">■ ■ ■ ■ ■ </a>

<ul>
  <li>■ ■ ■ ■ ■ </li>
  <li>■ ■ ■ ■ ■ ■ ■ ■ ■ ■ </li>
</ul>
```

تمرين الفصل

- اعمل صفحة فيها h1 و فقرتين و صورة و لسته و لينك.
- غير ال alt واقل الصور من ال DevTools وشوف بيظهر إيه.

بلاي ليست الفصل: أساسيات HTML

الرابط: _____

الفصل 7 — الفورمات و Semantic HTML

صفحة الناس تفهمها وجوجل كمان

Forms

- form وجواها input و textarea و select و button.
- أنواع input: text, email, password, number, date, checkbox, radio, file.
- بتسهل على اللي بيستخدم الصفحة input مربوطة بال label.
- بيضبطوا المدخلات من غير كود placeholder و required.

SEO وشوية Semantic HTML

- استخدم الوسم اللي معناه صح مش div في كل حاجة.
- لكل صفحة meta description و title.
- يوصف الصورة فعلا alt.
- العناوين بترتيب: h1 وبعده h2 وبعده h3.

الفورم من أول الكتابة للإرسال

- كل input لازم يبقى إيه name علشان قيمته تتبع، و label علشان المستخدم يعرف المطلوب. النوع المناسب زي email أو number.
- بيدي تحقق ولوحة مفاتيح مناسبة على الموبايل.
- لما الفورم يتبع، البيانات محتاجة مكان يستقبلها. في البداية ركز على الشكل والتحقق، وبعد ما تتعلم Backend هتخزنها فعلا.
- مستخدمش placeholder بدل label.
- اكتب رسالة خطأ مفهومة.
- اختبر الفورم بالكيبورد ومن الموبايل.

مثال: فورم تسجيل بسيطة

```
<form>
  <label for="email">■■■■■■■■■■</label>
  <input id="email" type="email" required
    placeholder="name@mail.com" />
  <button type="submit">■■■■■■■■■■</button>
</form>
```

تمرين الفصل

- اعمل فورم تواصل فيها اسم وإيميل ورسالة وزرار.
- حول كل div في صفحتك للوسم الصح (header, main, footer).

بلاي ليست الفصل: Semantic و HTML Forms

الرابط: _____

الفصل 8 – CSS – الشكل والألوان

من صفحة وحشة لصفحة تفتح النفس

إيه هو CSS؟

link هي اللي بتقول العنصر بيان إزاي: لونه، خطه، مسافته، حدوده، حجمه. بتكتبها في ملف لوحده وبتربطه بالصفحة بـ CSS في ال head.

- #id، و class، اسم العنصر، و Selectors.
- الوحدات: px, %, rem, em, vw, vh.
- margin وبعده، border وبعده، padding المحتوى، وبعده Box model.
- الألوان: بالاسم، أو hex، أو rgb، أو hsl.
- Specificity: مين اللي بيكسب لما قاعدتين يتخانقوا.

ليه التنسيق ساعات مش بيتطبق؟

المتصفح يجمع كل قواعد CSS ويحدد مين الأقوى حسب ال specificity والترتيب. لو قاعدتين بنفس القوة، اللي مكتوبة تحت هي اللي تكسب.

ابدأ بتنسيق عام لـ body، وبعدها الأقسام، وبعدها العناصر الصغيرة. استخدم classes واضحة بدل selectors طويلة صعب تراجعها.

- راجع إن ملف CSS مربوط صح.
- افتح Inspect وشوف القاعدة المشطوبة.
- استخدم box-sizing: border-box من البداية.

مثال: أول تكتيه CSS

```
body {
  font-family: system-ui, sans-serif;
  background: #faf8f4;
  color: #141414;
  padding: 24px;
}

.card {
  border: 2px solid #141414;
  border-radius: 12px;
  padding: 16px;
}
```

تمرين الفصل

- لون صفحتك واضبط الخط والمسافات.
- اعمل card. واستخدمها في 3 كروت.

بلاي ليست الفصل: أساسيات CSS

الرابط: _____

الفصل 9 — Grid و Flexbox

تقسيم الصفحة براحتك

Flexbox

- بتخلي العنصر حاوية مرنة `display: flex`.
- `flex-direction`, `justify-content`, `align-items`, `gap`, `flex-wrap`.
- أحسن حاجة لصف واحد أو عمود: ال navbar، كروت جنب بعض.

Grid

- للتقسيم بشبكة `grid-template-columns` مع `display: grid`.
- `repeat`, `fr`, `minmax`, `gap`.
- أحسن حاجة لتقسيم الصفحة كلها وللمعارض.

نصيحة

بلاش حفظ. افتح صفحة فاضية وجرب كل خاصية لوحدها وشوف بعينك بتعمل إيه — دي أسرع طريقة تثبت.

أختار مين؟

لو بتوزع عناصر في اتجاه واحد، صف أو عمود، استخدم Flexbox. لو عندك صفوف وأعمدة مع بعض، Grid غالباً أوضح. وممكن تستخدم الاثنين جوه بعض عادي.

جرب تحط border مؤقت على العناصر وانت بتقسم. الحدود هتوريك حجم كل صندوق والمسافة الحقيقية بدل التخمين.

- للمسافة بين العناصر gap
- مع الاتجاه الأساسي justify-content
- مع الاتجاه العكسي align-items

مثال: Flexbox لصف واحد

```
.navbar {
  display: flex;
  justify-content: space-between;
  align-items: center;
  gap: 16px;
}
```

مثال: Grid لتقسيم الصفحة

```
.projects {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 20px;
}
```

تمرين الفصل

- اعمل navbar ب flex فيه اسمك على ناحية ولينكات على الثانية.
- اعمل شبكة 3 أعمدة ب grid وشوف شكلها لما تصغر الشاشة.

بلاي ليست الفصل: Flexbox و Grid

الرابط: _____

الفصل 10 — الصفحة على الموبايل

أغلب الناس هتفتح من التليفون

الأساسيات

- من غيرها الموبايل هيصغر الصفحة — head في ال meta viewport
- ابدأ بالموبايل الأول وبعدين وسع للشاشة الكبيرة.
- Media queries: @media (min-width: 768px) { ... }
- خلي الصور مرنة بـ max-width: 100%
- جرب مقاسات مختلفة من ال DevTools.

اختبار حقيقي مش مجرد تصغير

التصميم المرن مش إن كل حاجة تصغر. ساعات القائمة تتغير، الأعمدة تبقى عمود واحد، والخط والمسافات يتظبطوا. ابدأ بأقل مساحة وبعدين أضف قواعد للشاشات الأكبر.

اختبر عرض ٣٢٠ و٧٦٨ و١٢٨٠ بكسل، وجرب محتوى طويل. الشكل اللي شغال مع اسم قصير ممكن ييوظ مع عنوان طويل.

- مفيش سحب أفقي في الصفحة.
- الأزرار سهلة للمس بالإصبع.
- النص مقروء من غير تكبير.

مثال: موبايل الأول وبعدين الشاشة الكبيرة

```
<meta name="viewport"
  content="width=device-width, initial-scale=1" />

.projects { grid-template-columns: 1fr; }

@media (min-width: 768px) {
  .projects { grid-template-columns: repeat(3, 1fr); }
}
```

تمرين الفصل

- افتح صفحتك على مقاس موبايل من DevTools وظبط كل حاجة مكسورة.
- خلي الشبكة عمود واحد على الموبايل و٣ أعمدة على اللابتوب.

بلاي ليست الفصل: Responsive Design

الرابط: _____

الفصل 11 — مشروع ١ — صفحتك الشخصية

أول حاجة هتنزلها وتبعثها لحد

المطلوب

- قسم تعريف: صورة واسم وسطر يقول انت مين.
- قسم مهاراتك في لسته.
- قسم مشاريعك: ٣ كروت بصورة وعنوان وسطرين.
- قسم تواصل بلينكاتك.
- تنسيق كامل بـ CSS وبيان حلو على الموبايل.

خطوة زيادة

نزلها مجاناً على GitHub Pages وابعت اللينك لحد. أول مرة حد يفتح حاجة عملتها إحساس مختلف تماماً.

قسم المشروع لمراحل

- اليوم الأول اكتب المحتوى وHTML، الثاني ضبط الألوان والمسافات، الثالث اعمل نسخة الموبايل، والرابع راجع وانشر. متحاولش تعمل كل حاجة مرة واحدة.
- قبل النشر راجع اللينكات والصور والإملاء، وافتح الصفحة من موبايل حقيقي. اكتب README صغير فيه صورة للمشروع واللينك المباشر.
- نسخة بسيطة مكتملة أحسن من تصميم ضخيم ناقص.
- استخدم محتواك الحقيقي.
- اطلب من شخص يجربها من غير ما تشرح له.

تمرين الفصل

- خالص الصفحة كاملة بأقسامها الأربعة.
- نزلها على GitHub Pages وابعت اللينك لحد تعرفه.

بلاي ليست الفصل: مشروع الصفحة الشخصية

الرابط: _____

الجزء الثالث

والحركة في الصفحة JavaScript

MOONASSER

الفصل 12 — JavaScript — الأساسيات

من هنا الصفحة تبدأ تتحرك

اللي محتاج تعرفه

- المتغيرات: let و const و var سيبها.
- الأنواع: string, number, boolean, null, undefined.
- المقارنات، واستخدم === مش ==.
- الشروط: if / else if / else و switch.
- التكرار: for و while و forEach.
- الدوال: function و arrow functions.
- Arrays و Objects، map و filter وحاجات مهمة زي find.

تتبع الكود على ورقة

لما تشوف كود جديد، امش سطر سطر واكتب قيمة كل متغير. الطريقة دي بتفهمك الشرط وال loop أسرع من حفظ شكلهم.

- استخدم const بشكل افتراضي، و let بس لو القيمة هتتغير. اختار أسماء بتقول القيمة بتعمل إيه زي totalPrice بدل x.
- قارن النوع والقيمة باستخدام ===.
- الدالة تعمل مهمة واحدة واضحة.
- اطبع القيم بـ console.log وقت التجربة.

مثال: متغيرات وشروط وتكرار ودوال

```
const name = "Mohamed";
let score = 0;

if (score === 0) {
  console.log("■■■■ ■■■■");
} else {
  console.log("■■■■■");
}

for (let i = 1; i <= 3; i++) console.log(i);

const double = (n) => n * 2;
console.log(double(5)); // 10
```

مثال: map و filter — هتستخدمهم كثير جدا

```
const nums = [1, 2, 3, 4, 5];
const evens = nums.filter((n) => n % 2 === 0); // [2,4]
const squares = nums.map((n) => n * n); // [1,4,9,16,25]
```

تمرين الفصل

- اكتب دالة بتاخذ رقم وترجع لو زوجي ولا فردي.
- استخدم map و filter على لسته أسماء.

بلاي ليست الفصل: أساسيات JavaScript

الرابط: _____

الفصل 13 — ال DOM — تعديل الصفحة بالكود

هنا بتحس إنك بتعمل حاجة حقيقية

DOM

- document.querySelector لأي عنصر
- لتغيير العنصر classList و innerHTML و textContent
- addEventListener مع click و input و submit للتعامل مع
- لو عايز تضيف عناصر جديدة append و createElement
- لو عايز تحفظ بيانات بسيطة في المتصفح localStorage

في سطرين Async

- لو عايز تجيب بيانات من النت fetch
- Promises و async / await
- هي الصيغة الي الدنيا كلها بتتبادل بيها البيانات JSON

اربط الحدث بالنتيجة

أي تفاعل غالبا فيه ٣ خطوات: اختار العنصر، اسمع الحدث، وبعدها غير البيانات أو الشكل. لو الزرار مش شغال، راجع كل خطوة لوحدها.

متخيلش البيانات الأساسية موجودة في شكل الصفحة بس. خزنها في array أو object، وبعدها اعمل render يعرضها. كده الإضافة والحذف هيبقوا أسهل.

- تأكد إن selector لقي العنصر.
- استخدم textContent للنص العادي.
- امنع إرسال الفورم الافتراضي ب preventDefault.

مثال: زرار يغير كلام في الصفحة

```
const btn = document.querySelector("#btn");
const out = document.querySelector("#out");

btn.addEventListener("click", () => {
  out.textContent = "■■■■■■■■!";
  out.classList.add("done");
});
```

مثال: جلب بيانات من API

```
async function load() {
  const res = await fetch("https://api.example.com/users");
  const data = await res.json();
  console.log(data);
}
load();
```

تمرين الفصل

- اعمل زرار يبدل بين الوضع الليلي والنهاري بـ `classList`.
- خذ اسم من `input` واعرضه في الصفحة بعد الضغط.

بلاي ليست الفصل: DOM و التفاعل

الرابط: _____

الفصل 14 — مشروع ٢ — تطبيق مهام

مشروع مستهلك، بس يعلم فعلا

المطلوب

- تضيف مهمة جديدة من `input`.
- تعلم عليها إنها خلصت وتقدر تمسحها.
- تحفظها في `localStorage` تفضل موجودة لو قفلت الصفحة.
- عداد للمهام اللي فاضلة.
- شكل نظيف ويشغل على الموبايل.

وبعد جرب

- تطبيق طقس بـ fetch من API مجاني.
- آلة حاسبة.
- معرض صور بفلتر.

شكل البيانات الأول

خلي كل مهمة object فيه id و text و done. لما المستخدم يضيف أو يمسح، عدل ال array، احفظها، وبعدها اعرض القائمة من جديد.

- اختبر الحالات اللي الناس بتنساها: input فاضي، مهمتين بنفس الاسم، إعادة تحميل الصفحة، ولسته طويلة.
- اعمل دوال منفصلة للإضافة والحذف والحفظ والعرض.
- متخزنش عناصر HTML نفسها.
- اظهر رسالة لما مفيش مهام.

مثال: قلب المشروع: إضافة مهمة وحفظها

```
let todos = JSON.parse(localStorage.getItem("todos") || "[]");

function add(text) {
  todos.push({ text, done: false });
  localStorage.setItem("todos", JSON.stringify(todos));
  render();
}
```

تمرين الفصل

- خلص الإضافة والحذف والتعليق إنها خلصت.
- حفظ في localStorage + عداد للمهام الفاضلة.

بلاي ليست الفصل: مشروع To-Do App

الرابط: _____

الجزء الرابع
المسارات: تختار إيه؟

MOONASSER

الفصل 15 — تختار مسارك إزاي؟

قبل ما تتوه في الأسماء الكثير

طريقة عملية

متخاترش من الكلام ولا من فيديو حد بيقول أحسن مسار. خد كل مسار أسبوع، اعمل فيه أصغر مشروع تقدر عليه، وشوف انت مستمتع فين — والأهم: مستحمل تكمل فيه بعد ما الحماس يهدى.

• بتحب تشوف النتيجة على الشاشة على طول؟ Frontend

• بتحب المنطق والبيانات؟ Backend

• بتحب حاجة بتتحرك في إيدك؟ Robotics

• بتحب الألعاب؟ Game Dev

• بتحب الأرقام والتوقعات؟ AI / Data

تجربة صغيرة قبل القرار

اعمل أسبوع تجريبي لكل مسار: يومين أساسيات، ٣ أيام مشروع صغير، ويوم مراجعة. سجل إيه اللي استمتعت بيه وإيه اللي ضايقتك بدل ما تعتمد على حماس أول فيديو.

سوق العمل مهم، بس قدرتك تكمل أهم. المسار اللي تقدر تمارس فيه شهر هو اللي هتتميز فيه.

• متبدأش مسارين كبار مع بعض.

• المبادئ الأساسية بتنفعك في أي مسار.

• القرار مش نهائي وتقدر تغيره.

تمرين الفصل

• اختار مسارين وجرب في كل واحد أصغر مشروع لمدة أسبوع.

• اكتب بعد الأسبوعين: استمتعت فين أكثر ولية.

بلاي ليست الفصل: اختيار المسار

الرابط: _____

الفصل 16 — Frontend

اللي المستخدم بيشفوفه ويلمسه

بتعمل إيه بالضبط؟

بتحول التصميم لواجهة شغالة، سريعة، ومفهومة، وتبان صح على أي مقاس شاشة.

- الأول: HTML, CSS, JavaScript, Git.
- بعدها: React (أو Vue) مع Tailwind CSS.
- مفاهيم: components, state, routing, performance.
- مشروع بداية: لوحة بسيطة بتقرأ بيانات من API.

بعد الأساسيات

قبل React لازم تكون مرتاح في JavaScript وال DOM. ال framework بينظم الشغل، لكنه مش بديل عن فهم المتصفح.

في أي مشروع اهتم بثلاث حاجات: شكل واضح، استخدام سهل، وتحميل سريع.

- قسم الواجهة لأجزاء صغيرة.
- خلي البيانات هي مصدر الحقيقة.
- اختبر لوحة المفاتيح والموبايل.

مثال: أول component في React

```
function Card({ title }) {  
  return <div className="card"><h3>{title}</h3></div>;  
}  
  
export default function App() {  
  return <Card title="MOHAMED NASSER" />;  
}
```

تمرين الفصل

- اعمل صفحة React فيها component واحد بيستقبل props.

بلاي ليست الفصل: مسار Frontend

الرابط: _____

Backend — 17 الفصل

اللي شغال ورا الستارة

بتعمل إيه بالضبط؟

بتبني ال API وقاعدة البيانات والمنطق: مين مسموح له يعمل إيه، والبيانات بتتخزن وترجع إزاي.

- اختار لغة واحدة: JavaScript/Node أو Python.
- قواعد بيانات: PostgreSQL (SQL) الأول، وبعدها NoSQL.
- مفاهيم: REST, authentication, validation, security.
- مشروع بداية: API لتطبيق مهام فيه تسجيل دخول.

طلب يدخل ونتيجة تخرج

المستخدم بيعت request، السيرفر يتحقق من البيانات والصلاحيات، يتعامل مع قاعدة البيانات، ويرجع response واضح.

متحطش الثقة في البيانات اللي جاية من المستخدم. اعمل validation لكل قيمة قبل الحفظ.

- استخدم status code مناسب.
- متطلعش تفاصيل الأخطاء السرية.
- خزن كلمات السر بطريقة آمنة، مش كنص عادي.

مثال: أصغر API ممكن بـ Node و Express

```
import express from "express";
const app = express();

app.get("/api/todos", (req, res) => {
  res.json([{ id: 1, text: "██████ Backend" }]);
});

app.listen(3000);
```

مثال: استعمال SQL بسيط

```
SELECT name, email
FROM users
WHERE created_at > '2026-01-01'
ORDER BY name;
```

تمرين الفصل

- اعمل API فيه route واحد بيرجع لسته JSON.

بلاي ليست الفصل: مسار Backend

الرابط: _____

الفصل 18 — Full-Stack

تشوف الصورة كلها

الفكرة

- مش معناها تتقن كل حاجة. معناها تعرف تبني منتج كامل لوحده: واجهة وسيرفر وقاعدة بيانات وتنزله Full-stack.
- خد Frontend لحد ما تكون مرتاح فيه، وبعدها Backend.
- اتعلم تنزل مشروعك (deployment) وتتعامل مع الـ env variables.
- أحسن مسار لو نفسك تبني مشاريعك بنفسك أو تشتغل freelance.

اربط القطع ببعض

- ابدأ بميزة واحدة كاملة: شاشة إضافة، API يستقبل، وجدول يحفظ. لما الدورة دي تشتغل زود باقي المميزات.
- اكتب شكل البيانات المتفق عليه بين الواجهة والسيرفر قبل التنفيذ.
- حالات التحميل والخطأ جزء من الميزة.
- افصل الإعدادات السرية عن الكود.
- جرب المشروع بعد النشر مش على جهازك بس.

تمرين الفصل

- اربط صفحة بسيطة بـ API انت عملته وشوف البيانات ظاهرة.

بلاي ليست الفصل: مسار Full-Stack

الرابط: _____

الفصل 19 — الموبايل

تطبيق في إيد الناس

الخيارات

- React Native — JavaScript لو مرتاح في
- Flutter (Dart) — شكل وأداء موحد على النظامين
- Native: Kotlin ل أندرويد ، Swift ل iOS.
- مفاهيم: navigation, state, permissions, offline storage.
- مشروع بداية: تطبيق ملاحظات ييحفظ على التليفون.

فكر كتطبيق موبايل

الموبايل فيه شاشة صغيرة، لمس، صلاحيات، وانقطاع إنترنت. صمم الرحلة على الأساس ده بدل ما تنقل موقع زي ما هو. ابدأ بشاشتين وتنقل بسيط وتخزين محلي قبل ما تدخل في خدمات كبيرة.

- احترم زر الرجوع.
- وضح ليه محتاج أي صلاحية.
- اختبر على جهاز حقيقي.

مثال: شاشة في React Native

```
import { View, Text } from "react-native";

export default function Home() {
  return (
    <View>
      <Text>■■■■■ ■■ ■■■■■■■■■■</Text>
    </View>
  );
}
```

تمرين الفصل

- اعمل تطبيق شاشة واحدة فيه اسمك وزرار.

بلاي ليست الفصل: مسار Mobile

الرابط: _____

الفصل 20 — الألعاب

أصعب وأحلى مسار في نفس الوقت

الطريق

- الأشهر للمبتدئين والمحترفين — C# مع Unity
- أساسيات: GameObjects, components, physics, prefabs, UI
- الرياضيات مهمة هنا: vectors, transforms, delta time
- أبدأ بلعبة صغيرة جداً وخلصها للآخر.
- الغلطة المشهورة: تبدأ بلعبة ضخمة وتسيبها بعد أسبوعين.

حلقة اللعبة

اللعبة بتكرر تحديث الحالة ورسم النتيجة مرات كتير في الثانية. الحركة لازم تعتمد على الوقت، مش سرعة الجهاز، علشان تبقى ثابتة.

أبدأ بميكانيكية واحدة ممتعة: حركة وقفز أو تصويب، وبعدها أضف النقاط والقائمة والصوت.

- اعمل نسخة قابلة للعب بسرعة.
- متبدأش بالجرافيك النهائي.
- اختبر الإحساس قبل زيادة المحتوى.

مثال: تحريك عنصر في Unity بـ C#

```
void Update() {  
    float x = Input.GetAxis("Horizontal");  
    transform.Translate(x * speed * Time.deltaTime, 0, 0);  
}
```

تمرين الفصل

- اعمل مكعب بيتحرك بالكيبورد في Unity.

بلاي ليست الفصل: مسار Game Dev / Unity

الرابط: _____

الفصل 21 — الروبوتات

كود يتحرك في الحقيقة

الطريق

- ESP32 وبعدها Arduino مع C++
- إلكترونيات أساسية: sensors, motors, drivers, power
- التحكم: PID وقرائة الحساسات وتنضيف الإشارة.
- بعد كده: ROS و OpenCV.
- مشروع بداية: عربية تمشي على خط أو تلف عن الحواجز.

الكود يتعامل مع عالم حقيقي

- الحساسات فيها ضوضاء والموتور مش دايما يتحرك بنفس الدقة. اقرأ القيمة كذا مرة، راقبها، وحط حدود أمان.
- اختبر كل قطعة لوحدها قبل تجميع الروبوت كله: حساس، بعدها موتور، بعدها القرار اللي بينهم.
- راجع مصدر الكهرباء.
 - استخدم Serial Monitor للتشخيص.
 - افصل الطاقة قبل تغيير التوصيلات.

مثال: أول كود Arduino — لمبة تور وتوم

```
void setup() { pinMode(13, OUTPUT); }

void loop() {
  digitalWrite(13, HIGH);
  delay(500);
  digitalWrite(13, LOW);
  delay(500);
}
```

تمرين الفصل

- وصل LED وخليه ينور وينوم بـ Arduino.

بلاي ليست الفصل: مسار Robotics

الرابط: _____

الفصل 22 — الأمن السيبراني

تفكر زي المهاجم وتدافع

الطريق

- أساسيات الشبكات و Linux و سطر الأوامر.
- وأشهر ثغرات المواقع OWASP Top 10
- أدوات: Burp Suite, Nmap, Wireshark.
- اتدرب على منصات زي TryHackMe و Hack The Box.
- مهم جدا: متجربش على حاجة مش بتاعتك ومن غير إذن.

اتعلم بشكل قانوني

- اعمل معمل على جهازك أو استخدم منصات تدريب مصممة للاختبار. الهدف تفهم الثغرة وإزاي تمنعها، مش تحفظ أداة.
- لكل تجربة اكتب: نقطة الضعف، تأثيرها، خطوات اكتشافها، وطريقة الإصلاح.
- إذن واضح قبل أي اختبار.
 - متحتفظش ببيانات حقيقية.
 - حدث الأدوات والأنظمة باستمرار.

تمرين الفصل

- خالص أول غرفة مجانية على TryHackMe و اكتب اللي فهمته.

بلاي ليست الفصل: مسار Cybersecurity

الرابط: _____

الفصل 23 — الذكاء الاصطناعي

أكثر حاجة مطلوبة دلوقتي

الطريق

- pandas و numpy كويس، ومعاها Python.
- رياضة أساسية: إحصاء واحتمالات وجبر خطي بسيط.
- الأول، والشبكات العصبية بعدين scikit-learn.
- بعدها PyTorch أو TensorFlow.
- مشروع بداية: تصنيف صور بسيط أو توقع أسعار من بيانات جاهزة.

البيانات قبل الموديل

جودة البيانات أهم من اختيار موديل معقد. نظف القيم الناقصة، افهم الأعمدة، واقسم البيانات لتدريب واختبار. قارن النتيجة بحل بسيط الأول. لو الموديل المعقد مش أحسن بوضوح، البسيط أسهل في الفهم والصيانة.

- متدريش وتختبر على نفس البيانات.
- اختار مقياس مناسب للمشكلة.
- راجع التحيز والخصوصية.

مثال: موديل تعلم آلة في ٦ سطور

```
from sklearn.tree import DecisionTreeClassifier

X = [[1, 2], [2, 3], [3, 1]]
y = [0, 1, 0]
model = DecisionTreeClassifier().fit(X, y)
print(model.predict([[2, 2]]))
```

تمرين الفصل

- درب موديل تصنيف بسيط على dataset جاهز من sklearn.

بلاي ليست الفصل: مسار AI / Machine Learning

الرابط: _____

الجزء الخامس

الأدوات والـ AI وخطة مذاكرة

MOONASSER

الفصل 24 — Git و GitHub

مش اختياري، دي جزء من الشغل

أوامر هتستخدمها كل يوم

- git init و git clone
- git status و git add و git commit -m
- git push و git pull
- git branch و git checkout -b و merge
- ملف .gitignore. علشان الملفات السرية ماتطلعش.

طيب ليه GitHub؟

حسابك على GitHub هو ال CV الحقيقي بتاعك. أي حد عايز يشوف انت بتعمل إيه هيفتحه. كل مشروع صغير تنزله بيبي السجل ده.

كل commit يحكي خطوة

اعمل commit صغير بعد ما جزء واضح يشتغل، واكتب رسالة تصف التغيير. كده لو حاجة باظت تعرف ترجع وتفهم التاريخ. قبل push راجع git status علشان متبعثش ملفات سرية أو ضخمة بالغلط.

- يشرح التشغيل README.
- استخدم branch للتجارب الكبيرة.
- حل التعارض بهدوء واقرا الجزئين.

مثال: الدورة اللي هتعيدها كل يوم

```
git init
git add .
git commit -m "first commit"
git branch -M main
git remote add origin <repo-url>
git push -u origin main
```

تمرين الفصل

- نزل مشروعك على GitHub وخلي فيه README بالعربي.
- اعمل branch جديد، غير حاجة، وادمجه في main.

بلاي ليست الفصل: Git و GitHub

الرابط: _____

الفصل 25 — أدوات الـ AI للمبرمج

خليها مساعد، مش بديل عنك

أدوات مفيدة

- شرح وتبسيط ومراجعة كود — ChatGPT / Claude
- ييكمولوا الكود وانت بتكتب — GitHub Copilot / Cursor
- تبني ييه تطبيق ويب كامل بالكلام — Lovable
- بحث ومعاة المصادر — Perplexity
- تشغيل موديلات على جهازك — Ollama

إزاي تستخدمها من غير ما تضر نفسك

- وانت بتتعليم: اطلب شرح، مش حل جاهز.
- لو مش فاهم الكود اللي جالك — بلاش تحطه.
- الـ AI بيغلط وهو واثق. راجع دايمًا.
- متكتبش أي مفاتيح أو بيانات سرية في المحادثة.

اسأل بطريقة تجيب نتيجة مفيدة

- ادي الأداة السياق: الهدف، اللغة، الكود الحالي، رسالة الخطأ، وإيه اللي جربته. واطلب منها تشرح السبب وتقتراح أقل تعديل.
- بعد ما تاخذ إجابة، اختبرها واقرا كل سطر. لو مش قادر تشرحه، متعتبرش المهمة خلصت.
- مترفعش أسرار أو بيانات عملاء.
 - اطلب مصادر للحاجات اللي ممكن تتغير.
 - استخدمه للمراجعة والتعلم، مش للنسخ الأعمى.

مثال: prompt وحش و prompt كويس

```
# bad prompt:
"build me a website"

# good prompt:
"an HTML + CSS page with a navbar and 3 project cards,
mobile first, no libraries, explain each section in 1 line."
```

تمرين الفصل

- اطلب من AI يشرح كود انت كاتبه — مش يكتب مكانك.
- اكتب prompt واضح ومفصل وقارن النتيجة بواحد مختصر.

الفصل 26 — خطة ٣ شهور واقعية

من الصفر لأول مشروع منشور

الشهر الأول

- أسبوع ١-٢: مفاهيم البرمجة + HTML.
- أسبوع ٣-٤: CSS و Flexbox والموبايل.
- آخر الشهر: صفحة شخصية منشورة.

الشهر الثاني

- أسبوع ٥-٦: أساسيات JavaScript.
- أسبوع ٧-٨: DOM ومشاريع صغيرة.
- آخر الشهر: تطبيق مهام + آلة حاسبة.

الشهر الثالث

- أسبوع ٩-١٠: Git و GitHub وتنظيم مشاريعك.
- أسبوع ١١-١٢: مشروع كبير واحد انت اللي تختاره.
- آخر الشهر: ٣ مشاريع على GitHub واختيار مسارك.

حاجة أخيرة

ساعة كل يوم أحسن بكثير من ٧ ساعات في يوم واحد وبعدها أسبوع نوم. الاستمرار هو الفرق كله.

قس التقدم بالحاجات الي خلصتها

- بدل ما تقول ذاكرت ٢٠ ساعة، عد المشاريع والميزات الي اشتغلت. كل أسبوع لازم يبقى فيه ناتج تقدر تفتحه أو تبعته.
- لو تأخرت، متضغطش أسبوعين في يوم. حرك الخطة وكمل من آخر نقطة؛ الاستمرار أهم من التاريخ المكتوب.
- هدف أسبوعي واحد واضح.
- يوم ثابت للمراجعة.
- انشر تقدمك وخذ رأي حقيقي.

تمرين الفصل

- احجز ساعة ثابتة كل يوم في التقويم من دلوقتي.
- اكتب اسم المشروع الكبير اللي هتعمله في الشهر التالت.

بلاي ليست الفصل: خطة ال 3 شهور

الرابط: _____

MOONASSER

خلاص كده؟ لأ — كده البداية

اللي بيفرق مش الكتاب ولا الكورس، اللي بيفرق إنك تفتح المحرر بكرة كمان.

Mohamed Nasser — moonasser.site

MOONASSER