TP Composants Bootstrap 5 : site du restaurant Dar El Medina

TP Composants Bootstrap 5 : site du restaurant Dar El Medina

1. Objectifs

- Ce TP de **4 heures** clôt le chapitre *Les composants Jumbotron, Cards et Carousels*. En le réalisant, vous allez apprendre à :
  - identifier et utiliser les **composants Bootstrap** (navbar, boutons, badges, alertes) et leurs attributs `data-bs-*`;
  - créer un **Jumbotron** en Bootstrap 5, où la classe `.jumbotron` n'existe plus, à l'aide des classes utilitaires ;
  - construire des **Cards** (image, corps, en-tête, pied, liste) et les disposer dans une grille responsive de hauteur égale ;
  - mettre en place un **Carousel** avec indicateurs, contrôles, légendes et défilement automatique ;
  - ouvrir des fenêtres **modales** (formulaire de réservation, détail d'un plat) et remplir leur contenu en JavaScript ;
  - générer les cartes à partir d'un fichier **JSON** chargé avec `fetch()`;
  - versionner son travail avec **Git**, le publier sur **GitHub** dès le départ et mettre le site en ligne avec **GitHub Pages**.
- **Prérequis :** HTML5, CSS3, système de grille Bootstrap (`container`, `row`, `col-*`), bases de JavaScript (sélection d'éléments, événements).
- **Avant la séance :** un compte **GitHub** gratuit, **Git** installé (vérifier avec `git --version`) et VS Code avec l'extension **Live Server**.

1. Description

- Le restaurant traditionnel **Dar El Medina** souhaite un site vitrine d'une seule page : présentation, menu, galerie photos, réservation en ligne et contact.
- La page finale est organisée en **sections** reliées par la barre de navigation. Chaque section mobilise un composant du chapitre :

apcpedagogie.com

Section de la page	Ancre	Composants principaux du chapitre
Barre de navigation	–	`navbar`, `navbar-toggler`, `collapse`, boutons
Accueil	`#accueil`	Jumbotron (utilitaires `p-5`, `bg-body-tertiary`, `display-*`, `lead`)
Notre menu	`#menu`	`card`, `row-cols-*`, badge, générées depuis JSON
Galerie	`#galerie`	`carousel`, indicateurs, contrôles, légendes
Réservation et détail d'un plat	fenêtres modales	`modal`, formulaire, événement `show.bs.modal`
Contact et pied de page	`#contact`	cards horizontales, utilitaires

- **Images :** pas besoin de télécharger d'images. Utilisez des images d'exemple de la forme `https://picsum.photos/seed/picsum/200/300` (par exemple `https://picsum.photos/seed/couscous/600/400`) : le même mot donne toujours la même image.
- **Répartition du temps :**

Partie	Composant(s) travaillé(s)	Durée
Partie 00 – Mise en place du projet et du dépôt GitHub	Git, GitHub	20 min
Partie 01 – Les composants Bootstrap	`navbar`, boutons, badges, alertes	20 min
Partie 02 – Créer un Jumbotron	utilitaires d'espacement, de couleur et de typographie	30 min
Partie 03 – Le composant Card	`card`, `row-cols-*`, `h-100`	40 min
Partie 04 – Le composant Carousel	`carousel`	35 min
Partie 05 – Le composant Modal	`modal`, `show.bs.modal`	35 min
Partie 06 – Cartes générées depuis un fichier JSON	`fetch()`, gabarits JavaScript	30 min
Partie 07 – Intégration finale et mise en ligne	tous + GitHub Pages, `git tag`	30 min
Total		**4 h**

apcpedagogie.com

1. Travail à réaliser

- **Partie 00Mise en place du projet et du dépôt GitHub**

- **Q0.1.** Créer le dossier `dar-el-medina` et son arborescence (à gauche), puis l'ouvrir dans VS Code.
- **Q0.2.** Dans `index.html`, écrire le squelette ci-dessous (à droite). Il charge Bootstrap 5.3 et Bootstrap Icons depuis un CDN, puis `bootstrap.bundle.min.js` en fin de `body`. Pourquoi faut-il la version `bundle` pour que les composants interactifs (navbar repliable, carousel, modal) fonctionnent ?
- **Q0.3.** Ouvrir la page avec **Live Server** (clic droit → *Open with Live Server*) et vérifier dans la console du navigateur (F12) qu'aucune erreur de chargement n'apparaît.

```
dar-el-medina/
├── index.html
├── css/
│   └── style.css
├── js/
│   └── main.js
├── data/
│   └── plats.json      (Partie 06)
├── captures/           (Partie 07)
├── README.md           (Partie 07)
└── reponses.md         (au fil du TP)
```

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Dar El Medina — Restaurant traditionnel</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css"
rel="stylesheet">
  <link href="css/style.css" rel="stylesheet">
```

apcpedagogie.com

```
</head>
<body>
  <!-- Partie 01 : barre de navigation -->
  <!-- Partie 02 : jumbotron (#accueil) -->
  <!-- Partie 03 : cartes du menu (#menu) -->
  <!-- Partie 04 : carrousel (#galerie) -->
  <!-- Partie 05 : fenêtres modales -->
  <!-- Partie 07 : contact et pied de page -->

  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
  <script src="js/main.js"></script>
</body>
</html>
```

- **Publier le projet sur GitHub dès le départ**

- **Q0.4.** Sur **github.com/new**, créer un dépôt **public** nommé `dar-el-medina`, **sans** cocher « Add a README », « .gitignore » ni « license ». Le dépôt doit être public pour la mise en ligne avec GitHub Pages en partie 07.
- **Q0.5.** Configurer Git une seule fois sur le poste avec votre nom et l'adresse e-mail de votre compte GitHub (commandes 1 et 2 ci-dessous).
- **Q0.6.** Dans le dossier `dar-el-medina`, initialiser le dépôt local, faire le premier *commit* et l'envoyer sur GitHub (commandes 3 à 8). Remplacer `VOTRE-COMPTE` par votre nom d'utilisateur GitHub.

```
# 1-2. Identité (une seule fois par poste)
git config --global user.name "Prénom Nom"
git config --global user.email "adresse@exemple.com"

# 3-8. Premier envoi du projet
cd dar-el-medina
git init
git add .
git commit -m "Partie 00 : structure du projet et squelette Bootstrap 5"
git branch -M main
git remote add origin https://github.com/VOTRE-COMPTE/dar-el-medina.git
git push -u origin main
```

apcpedagogie.com

```
# Vérifications utiles
git status          # fichiers modifiés
git log --oneline   # historique des commits
git remote -v       # adresse du dépôt GitHub
```

- **Q0.7.** Actualiser la page du dépôt sur GitHub et vérifier que `index.html` et les dossiers `css/` et `js/` apparaissent. Git n'envoie pas les dossiers vides : comment le constatez-vous pour `data/` et `captures/`? Envoyer le lien du dépôt à l'enseignant **avant de commencer la partie 01**.

Authentification : lors du premier `git push`, GitHub n'accepte plus le mot de passe du compte. Connectez-vous par le navigateur quand Git le propose (Git Credential Manager, VS Code ou GitHub Desktop), ou utilisez un *Personal Access Token* (*Settings → Developer settings → Personal access tokens*) à la place du mot de passe.
Règle du TP : un commit et un push à la fin de chaque partie.
Chaque partie se termine par un encadré « Git » qui donne le message de *commit* attendu. L'historique des commits fait partie de la note : un dépôt qui ne contient qu'un seul commit déposé en fin de séance n'obtient aucun point sur le critère Git.
Question de réflexion : pourquoi est-il préférable de faire plusieurs petits commits plutôt qu'un seul gros commit à la fin ?

- **Partie 01Les composants Bootstrap**

- **Objectif :** découvrir ce qu'est un composant Bootstrap et construire la barre de navigation du site.
- **Q1.1.** Dans `reponses.md`, expliquer la différence entre une **classe utilitaire** (ex. `mt-3`, `text-center`) et un **composant** (ex. `navbar`, `card`). Citer trois composants qui ont besoin de JavaScript pour fonctionner et deux qui n'en ont pas besoin.
- **Q1.2.** Créer une `nav` de classe `navbar navbar-expand-lg` en haut de la page, avec `data-bs-theme="dark"` et un fond `bg-dark`, contenant :
  - à gauche, une marque `navbar-brand`: une icône `bi bi-shop` suivie du texte « Dar El Medina » ;
  - des liens `nav-link` vers les ancres `#accueil`, `#menu`, `#galerie` et `#contact`;
  - à droite (`ms-auto`), un bouton `btn btn-warning` « Réserver » (il ouvrira une modale en partie 05).
- **Q1.3.** Ajouter le bouton `navbar-toggler` et envelopper les liens dans un `div.collapse.navbar-collapse` avec un `id`. Réduire la fenêtre sous 992 px : que se passe-t-il ? À quoi servent les

apcpedagogie.com

attributs `data-bs-toggle` et `data-bs-target` du bouton ?

- **Q1.4.** Remplacer `navbar-expand-lg` par `navbar-expand-md` puis par `navbar-expand-sm`. Quel est l'effet sur le point de rupture du menu ? Conserver `navbar-expand-lg`.
- **Q1.5.** Rendre la barre fixe en haut de l'écran avec `sticky-top`. Quelle est la différence avec `fixed-top` sur le contenu situé en dessous ?
- **Q1.6.** Sous la barre de navigation, ajouter une **alerte** `alert alert-info alert-dismissible fade show`: « Nouveau : réservez votre table en ligne ! » avec un bouton de fermeture `btn-close` (`data-bs-dismiss="alert"`). Ajouter un **badge** `badge text-bg-danger` « Nouveau » dans le texte de l'alerte.

Git – fin de la partie 01 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 01 : composants Bootstrap — navbar, alerte et badge"
git push
```

- **Partie 02Créer un Jumbotron Bootstrap**

- **Objectif :** créer la grande bannière d'accueil du site.
- **Q2.1.** La classe `.jumbotron` de Bootstrap 4 a été supprimée en Bootstrap 5. Ajouter-la à une `div` et observer le résultat. Pourquoi, selon vous, l'équipe de Bootstrap l'a-t-elle retirée ?
- **Q2.2.** Recréer le Jumbotron avec les classes utilitaires, sur le modèle ci-dessous (à gauche) : titre « Bienvenue chez Dar El Medina », texte d'accroche et bouton « Découvrir le menu » qui renvoie à `#menu`.
- **Q2.3.** Expliquer dans `reponses.md` le rôle de chaque classe : `p-5`, `mb-4`, `bg-body-tertiary`, `rounded-3`, `py-5`, `display-5`, `fw-bold`, `fs-4`, `col-md-8`, `btn-lg`.

```
<section id="accueil" class="p-5 mb-4 bg-body-tertiary rounded-3">
  <div class="container-fluid py-5">
    <h1 class="display-5 fw-bold">...</h1>
    <p class="col-md-8 fs-4">...</p>
    <a href="#menu" class="btn btn-primary btn-lg">...</a>
  </div>
</section>
```

apcpedagogie.com

```
/* css/style.css — Q2.5 */
.hero {
  background: linear-gradient(rgba(0,0,0,.55), rgba(0,0,0,.55)),
              url("https://picsum.photos/seed/medina/1600/700")
              center / cover no-repeat;
  min-height: 70vh;
}
```

- **Q2.4.** Remplacer `display-5` par `display-1`, puis `display-3`. Comment la taille du titre évolue-t-elle sur mobile ? Garder la valeur la plus lisible sur les deux formats.
- **Q2.5.** Transformer le Jumbotron en bannière avec image de fond : ajouter la classe `hero` (CSS ci-dessus, à droite) et remplacer `bg-body-tertiary` par `text-white`. À quoi sert le dégradé noir semi-transparent placé avant l'image ?
- **Q2.6.** Centrer verticalement et horizontalement le contenu de la bannière avec les utilitaires flexbox `d-flex`, `align-items-center` et `text-center`.
- **Q2.7.** Ajouter sous le bouton principal un second bouton `btn btn-outline-light btn-lg` « Réserver une table ». Sur mobile, les deux boutons doivent s'empiler et occuper toute la largeur : utiliser `d-grid gap-2 d-sm-flex justify-content-sm-center` sur leur conteneur.

Git – fin de la partie 02 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 02 : Jumbotron avec utilitaires et image de fond"
git push
```

- **Partie 03Le composant Card**

- **Objectif :** présenter les plats du restaurant sous forme de cartes dans une grille responsive.
- **Q3.1.** Créer la section `#menu` avec un titre « Notre menu ». Y placer **une seule** carte pour le plat « Couscous au poisson » (modèle ci-dessous) : image `card-img-top`, titre `card-title`, description `card-text` et bouton « Voir le détail ».
- **Q3.2.** Dans le pied de carte (`card-footer`), afficher le prix « 28 DT » à gauche et un badge `text-bg-success` « Nouveau » à droite (`d-flex justify-content-between`).

apcpedagogie.com

```
<div class="card h-100">
  <img src="https://picsum.photos/seed/couscous/600/400"
       class="card-img-top" alt="Couscous au poisson">
  <div class="card-body">
    <h5 class="card-title">...</h5>
    <p class="card-text">...</p>
  </div>
  <div class="card-footer">...</div>
</div>
```

- **Q3.3.** Ajouter cinq autres cartes (brik à l'œuf, ojja merguez, lablabi, makroudh, thé à la menthe). Placer les six cartes dans une grille `row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4`, chaque carte dans un `div.col`. Combien de cartes par ligne obtenez-vous sur mobile, tablette et ordinateur ?
- **Q3.4.** Donner des descriptions de longueurs différentes aux cartes. Que remarquez-vous sur leur hauteur ? Corriger avec la classe `h-100` sur chaque `.card` et expliquer pourquoi le pied de carte reste aligné.
- **Q3.5.** Fixer la même hauteur pour toutes les images (CSS : `height: 200px; object-fit: cover;`). Que se passe-t-il sans `object-fit: cover` ?
- **Q3.6.** Remplacer temporairement la grille par un `div.card-group`. Quelle différence visuelle observez-vous avec la grille `row-cols-*` ? Laquelle est la plus adaptée au mobile ? Revenir à la grille.
- **Q3.7.** Sur la carte « Thé à la menthe », ajouter un en-tête `card-header` « Boisson » et remplacer le texte par une liste `list-group list-group-flush` de trois lignes : « Thé vert », « Menthe fraîche », « Pignons ».
- **Q3.8.** Créer sous la grille une carte « Plat du jour » **horizontale** : image à gauche, texte à droite, avec `row g-0`, `col-md-4` et `col-md-8`. Utiliser une carte colorée `text-bg-warning`.

Question de réflexion :
Pourquoi place-t-on chaque carte dans une colonne `.col` au lieu de mettre directement les cartes dans la `.row` ? Que se passe-t-il pour l'espacement `g-4` sans ces colonnes ?
Git – fin de la partie 03 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 03 : cards du menu en grille responsive"
git push
```

apcpedagogie.com

- **Partie 04Le composant Carousel Bootstrap**

- **Objectif :** créer une galerie photos animée du restaurant.
- **Q4.1.** Créer la section `#galerie` et y placer un carrousel `carouselGalerie` de **quatre** diapositives (salle, terrasse, cuisine, desserts) sur le modèle ci-dessous. Une seule diapositive doit porter la classe `active`: que se passe-t-il si vous l'oubliez ?

```
<div id="carouselGalerie" class="carousel slide" data-bs-ride="carousel">
  <div class="carousel-indicators">
    <button type="button" data-bs-target="#carouselGalerie"
            data-bs-slide-to="0" class="active" aria-current="true"
            aria-label="Diapositive 1"></button>
    <!-- ... -->
  </div>
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img src="https://picsum.photos/seed/salle/1200/500"
           class="d-block w-100" alt="...">
      <div class="carousel-caption d-none d-md-block">...</div>
    </div>
    <!-- ... -->
  </div>
  <!-- boutons .carousel-control-prev et .carousel-control-next -->
</div>
```

- **Q4.2.** Ajouter les boutons « précédent » et « suivant » (`carousel-control-prev` / `carousel-control-next`) avec leurs icônes et un texte `visually-hidden`. À qui s'adresse ce texte invisible ?
- **Q4.3.** Compléter les quatre **indicateurs** (`carousel-indicators`). Vérifier que `data-bs-slide-to` commence à 0 et que l'`id` du carrousel est bien repris dans chaque `data-bs-target`.
- **Q4.4.** Ajouter une légende (`carousel-caption`) à chaque diapositive : un titre `h5` et une courte phrase. Pourquoi la classe `d-none d-md-block` cache-t-elle la légende sur mobile ?
- **Q4.5.** Régler la durée d'affichage à 3 secondes avec `data-bs-interval="3000"` sur chaque `carousel-item`, puis tester `data-bs-ride="false"`. Quelle est la différence entre les valeurs `carousel` et `false` de `data-bs-ride`?
- **Q4.6.** Remplacer l'effet de glissement par un fondu avec la classe `carousel-fade`.
- **Q4.7.** Les photos n'ont pas toutes la même proportion : fixer une hauteur de 450 px aux images du carrousel (CSS) avec `object-fit: cover`, et 250 px sur mobile grâce à une *media*

apcpedagogie.com

query.

- **Q4.8.** Copier le carrousel une seconde fois dans la page **sans changer son `id`**. Cliquer sur les flèches du second carrousel : que se passe-t-il ? Expliquer, puis supprimer la copie.

Git – fin de la partie 04 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 04 : carousel de la galerie avec légendes et fondu"
git push
```

- **Partie 05Le composant modal Bootstrap 5**

- **Objectif :** afficher un formulaire de réservation et le détail d'un plat dans des fenêtres modales.

- **Modale de réservation**

- **Q5.1.** Créer avant le `<script>` de fin de page une modale `modalReservation` composée de `modal-dialog`, `modal-content`, `modal-header` (titre + `btn-close`), `modal-body` et `modal-footer`.
- **Q5.2.** Relier les boutons « Réserver » de la navbar (Q1.2) et du Jumbotron (Q2.7) à cette modale avec `data-bs-toggle="modal"` et `data-bs-target="#modalReservation"`.
- **Q5.3.** Dans le corps de la modale, créer un formulaire Bootstrap : nom (`form-control`), téléphone, date (`type="date"`), heure, nombre de personnes (`form-select` de 1 à 8) et une case « Terrasse » (`form-check`). Disposer date et heure côte à côte avec `row g-3`.
- **Q5.4.** Dans le pied, ajouter un bouton « Annuler » (`data-bs-dismiss="modal"`) et un bouton « Confirmer » de type `submit` relié au formulaire par l'attribut `form`.
- **Q5.5.** Tester successivement les classes `modal-dialog-centered`, `modal-lg` et `modal-dialog-scrollable`, puis l'attribut `data-bs-backdrop="static"`. Décrire l'effet de chacun. Lesquels conservez-vous pour un formulaire de réservation, et pourquoi ?

- **Modale de détail d'un plat**

- **Q5.6.** Créer **une seule** modale `modalPlat` (taille `modal-lg`) contenant une image, une description et un prix vides.
- **Q5.7.** Sur chaque bouton « Voir le détail » des cartes, ajouter `data-bs-toggle="modal"`, `data-bs-target="#modalPlat"` et des attributs `data-nom`, `data-image`, `data-description`, `data-prix`.

apcpedagogie.com

- **Q5.8.** Dans `js/main.js`, compléter le code ci-dessous pour remplir la modale avec les informations du plat cliqué. Pourquoi est-il préférable d'avoir une modale réutilisée plutôt que six modales écrites à la main ?

```
const modalPlat = document.getElementById('modalPlat');

modalPlat.addEventListener('show.bs.modal', (event) => {
  const bouton = event.relatedTarget;           // bouton cliqué
  const nom = bouton.getAttribute('data-nom');  // attribut data-nom
  modalPlat.querySelector('.modal-title').textContent = nom;
  // à compléter : image, description, prix
});
```

Question de réflexion :
Quelle est la différence entre les événements `show.bs.modal` et `shown.bs.modal`? Dans quel cas utiliseriez-vous le second (indice : placer le curseur dans le premier champ du formulaire) ?
Git – fin de la partie 05 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 05 : modales de réservation et de détail des plats"
git push
```

- **Partie 06Cartes générées depuis un fichier JSON**

- **Objectif :** ne plus écrire les cartes à la main, mais les générer à partir d'un fichier de données.
- **Q6.1.** Créer `data/plats.json` sur le modèle ci-dessous (à gauche) et y décrire les six plats de la partie 03, avec au moins deux plats `"nouveau": true`. Vérifier la validité du fichier avec un validateur JSON en ligne : quelles erreurs courantes rendent un JSON invalide ?
- **Q6.2.** Dans `index.html`, remplacer les six cartes écrites à la main par une grille vide `<div id="liste-plats" class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4"></div>`.
- **Q6.3.** Dans `js/main.js`, charger le fichier avec `fetch()` et générer une carte par plat avec un gabarit de chaîne (*template literal*) en reprenant exactement la structure HTML de la partie 03 (code à droite).

apcpedagogie.com

```
[
  {
    "id": 1,
    "nom": "Couscous au poisson",
    "categorie": "Plats",
    "prix": 28,
    "image": "https://picsum.photos/seed/couscous/600/400",
    "description": "Semoule fine, mérou et légumes de saison.",
    "nouveau": true
  },
  {
    "id": 2,
    "nom": "Brik à l'œuf",
    "categorie": "Entrées",
    "prix": 6,
    "image": "https://picsum.photos/seed/brik/600/400",
    "description": "Feuille de malsouka croustillante, œuf et thon.",
    "nouveau": false
  }
]
```

```
const liste = document.getElementById('liste-plats');

fetch('data/plats.json')
  .then((reponse) => reponse.json())
  .then((plats) => {
    liste.innerHTML = plats.map((plat) => `
      <div class="col">
        <!-- à compléter : la carte du plat -->
      </div>`).join('');
  })
  .catch((erreur) => {
    // à compléter : afficher une alerte Bootstrap .alert-danger
  });
```

- **Q6.4.** Afficher le badge « Nouveau » uniquement pour les plats où `nouveau` vaut `true` (opérateur ternaire dans le gabarit).
- **Q6.5.** Remplir les attributs `data-*` des boutons « Voir le détail » à partir du JSON. Vérifier que la modale de la partie 05 fonctionne toujours avec les cartes générées.

apcpedagogie.com

- **Q6.6.** Renommer temporairement le fichier JSON : afficher à la place de la grille une alerte `alert alert-danger` « Impossible de charger le menu ». Remettre ensuite le bon nom.
- **Q6.7.** Ouvrir `index.html` en double-cliquant dessus (adresse `file://`) au lieu de Live Server. Les cartes s'affichent-elles ? Expliquer pourquoi `fetch()` a besoin d'un serveur web.
- **Q6.8.** Au-dessus de la grille, ajouter un groupe de boutons `btn-group` « Tous », « Entrées », « Plats », « Desserts », « Boissons » qui filtre les cartes affichées selon `categorie`. Le bouton actif porte la classe `active`.

Question de réflexion :
Quel avantage le fichier `plats.json` apporte-t-il au restaurant quand il change sa carte ou ses prix, par rapport aux cartes écrites directement dans le HTML ?
Git – fin de la partie 06 : enregistrer et publier votre travail, puis vérifier le nouveau commit sur GitHub.

```
git add .
git commit -m "Partie 06 : cartes générées depuis plats.json avec fetch"
git push
```

- **Partie 07Intégration finale et mise en ligne**

- **Objectif :** terminer la page, la vérifier, puis la publier sur Internet avec GitHub Pages.
- **Q7.1.** Créer la section `#contact` avec trois cartes horizontales côte à côte (`row-cols-1 row-cols-md-3`) : adresse (`bi-geo-alt`), téléphone (`bi-telephone`) et horaires (`bi-clock`).
- **Q7.2.** Ajouter un pied de page sombre (`bg-dark text-white py-4`) avec le nom du restaurant, des icônes de réseaux sociaux et l'année.
- **Q7.3.** Vérifier que chaque lien de la navbar mène à la bonne section, et que tout s'affiche correctement à 375 px, 768 px et 1280 px avec l'outil « mode appareil » du navigateur (Ctrl+Maj+M).
- **Q7.4.** Valider `index.html` sur **validator.w3.org** et corriger les erreurs signalées. Vérifier aussi que chaque image a un attribut `alt` pertinent.
- **Q7.5.** Faire une capture de la page à chaque taille de la Q7.3 et les enregistrer dans le dossier `captures/`.

- **Finaliser le dépôt et mettre le site en ligne**

- **Q7.6.** Créer à la racine du projet un fichier `README.md` qui présente le site (modèle ci-dessous, à gauche) : description, lien de la démo en ligne, composants utilisés et captures d'écran.

apcpedagogie.com

- **Q7.7.** Vérifier que `reponses.md` est complet, puis faire le dernier commit et marquer la version finale avec l'étiquette `v1.0` (commandes à droite).
- **Q7.8.** Sur GitHub, ouvrir *Settings* → *Pages*. Dans *Build and deployment*, choisir *Deploy from a branch*, la branche `main` et le dossier `/ (root)`, puis enregistrer. Après une à deux minutes, ouvrir l'adresse `https://VOTRE-COMPTE.github.io/dar-el-medina/` et vérifier que le site s'affiche, cartes JSON comprises.
- **Q7.9.** Ajouter le lien du site en ligne dans le `README.md` et dans la zone *About* du dépôt (icône ⚙ à droite de la page du dépôt), puis faire un dernier commit et un push.

```
# Dar El Medina

Site vitrine d'un restaurant traditionnel réalisé avec
Bootstrap 5 dans le cadre du TP « Composants Bootstrap ».

## Démo en ligne
https://VOTRE-COMPTE.github.io/dar-el-medina/

## Composants utilisés
- Navbar responsive
- Jumbotron (utilitaires Bootstrap 5)
- Cards en grille responsive
- Carousel avec indicateurs et légendes
- Modals : réservation et détail d'un plat
- Données des plats chargées depuis data/plats.json

## Captures d'écran
![Accueil](captures/accueil.png)
![Menu](captures/menu.png)

## Auteur
Prénom Nom — Groupe
```

```
git add .
git commit -m "Partie 07 : intégration finale, README et captures"
git push

git tag v1.0
git push origin v1.0
```

apcpedagogie.com

Question de réflexion :

Pourquoi le site fonctionne-t-il sur GitHub Pages alors qu'il ne fonctionnait pas en double-cliquant sur `index.html` (Q6.7) ? Pourquoi les noms de fichiers doivent-ils respecter les majuscules et minuscules une fois en ligne ?

1. Exercices complémentaires (bonus)

- **Mode sombre :** ajouter dans la navbar un bouton qui bascule l'attribut `data-bs-theme` de la balise `<html>` entre `light` et `dark`, et mémoriser le choix avec `localStorage`.
- **Confirmation :** après l'envoi du formulaire de réservation, fermer la modale et afficher un **toast** Bootstrap « Réservation enregistrée ».
- **Foire aux questions :** ajouter une section FAQ avec le composant `accordion` (parking, plats végétariens, réservation de groupe).
- **Navigation active :** activer le `scrollspy` sur la page pour que le lien de la section visible soit surligné dans la navbar.

1. Compte rendu

- **À rendre :**
  - **le lien de votre dépôt GitHub** `dar-el-medina`, envoyé dès la partie 00 : c'est ce dépôt qui est corrigé, dans l'état de l'étiquette `v1.0`;
  - **le lien du site en ligne** sur GitHub Pages ;
  - dans le dépôt, le fichier `reponses.md`: réponses aux questions Q0.2, Q0.7, Q1.1, Q1.3, Q1.5, Q2.1, Q2.3, Q2.5, Q3.3, Q3.4, Q3.6, Q4.1, Q4.5, Q4.8, Q5.5, Q5.8, Q6.1, Q6.7 et aux questions de réflexion ;
  - dans le dépôt, le `README.md` et les captures d'écran du dossier `captures/`.
- **Barème indicatif (/20) :**

Critère	Points
Partie 00 – arborescence, squelette Bootstrap 5 et page sans erreur	1
Partie 01 – navbar responsive, alerte et badge	1,5
Partie 02 – Jumbotron avec utilitaires et image de fond	2
Partie 03 – cards : grille responsive, hauteur égale, en-tête, liste, carte horizontale	3,5
Partie 04 – carousel : indicateurs, contrôles, légendes, fondu, images uniformes	2,5

apcpedagogie.com

Critère	Points
Partie 05 – modales : formulaire de réservation et détail dynamique d'un plat	3
Partie 06 – cartes générées depuis JSON, gestion d'erreur et filtre par catégorie	2,5
Partie 07 – contact, pied de page, validation W3C et site en ligne sur GitHub Pages	2,5
Git / GitHub – dépôt créé en partie 00, un commit par partie avec un message clair, `README.md`, étiquette `v1.0`	1,5
Total	**20**
Bonus	+2 max

Question de réflexion (à rédiger en 5 lignes) :

Quels sont les avantages et les limites d'un framework comme Bootstrap par rapport à un site entièrement écrit en CSS « à la main » ?

Source : https://apcpedagogie.com/tp-composants-bootstrap-5-site-du-restaurant-dar-el-medina · téléchargé le 07/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com