Bootstrap Grid System

Slide 1: What is Bootstrap Grid?

- Bootstrap Grid is a **responsive layout system**
- It is based on **12 columns per row**
- Uses **Flexbox internally**
- Helps us design layouts for **mobile, tablet, laptop, desktop**

  Goal: Same website → works on all screen sizes

Slide 2: Core Structure (Always Remember)

Every Bootstrap grid follows this structure:

- `.container` → page padding + center
- `.row` → activates grid (flexbox)
- `.col-*` → columns

Example structure:

```
container
 └── row
     └── col
```

Slide 3: Why 12 Columns?

- 12 is easily divisible
- Common layouts:
- 12 → full width
- 6 + 6 → 2 columns
- 4 + 4 + 4 → 3 columns
- 3 + 3 + 3 + 3 → 4 columns

  Rule: **Total per row = 12**

Slide 4: Mobile-First Concept

Bootstrap is **mobile-first**

Meaning: - Small screens first - Bigger screens override smaller ones

Flow:

```
Mobile → Tablet → Laptop → Desktop
```

> Bigger screens inherit smaller screen rules

Slide 5: Breakpoints Explained

Class	Screen
col	all screens
col-sm-*	≥ 576px
col-md-*	≥ 768px
col-lg-*	≥ 992px
col-xl-*	≥ 1200px

Slide 6: Example – Responsive Columns

Requirement: - Mobile → 1 column - Tablet → 2 columns - Desktop → 3 columns

Column class used:

```
col-sm-12 col-md-6 col-lg-4
```

Logic: - 12 ÷ 12 = 1 - 12 ÷ 6 = 2 - 12 ÷ 4 = 3

Slide 7: Why `col-md-4` Works on Mobile?

If mobile class is NOT defined:

```
col-md-4
```

Then: - Mobile uses **same rule** - Because Bootstrap goes **downwards**

> Undefined breakpoint = inherits previous

Slide 8: `col-4` on Mobile

```
col-4
```

Meaning: - Applies on **all screen sizes** - Mobile included

Result: - 3 columns side by side even on phone

Slide 9: What is `.col` (No number)?

```
<div class="col"></div>
```

Behavior: - Auto width - All columns share space equally

Example: - 3 `.col` → each gets 4 columns

Slide 10: Why Columns Stay Side by Side on Mobile?

Because: - `.row` uses **Flexbox** - Flexbox default is `flex-direction: row` - Columns only stack when width becomes 12

Slide 11: Cards + Grid (Real Use Case)

We usually place **cards inside columns**

Pattern:

```
container
 └── row
```

```
      └── col
          └── card
```

This is professional & real-world layout

Slide 12: Why `h-100` is Used in Cards

```
<div class="card h-100">
```

Reason: - Makes card take **full column height** - Keeps all cards **same height** - Buttons align properly

---

Slide 13: Why Other Cards Grow in Height?

- `.row` is flexbox
- Tallest column defines row height
- Other columns stretch automatically
- `h-100` forces cards to fill that height

  Equal height is intentional behavior

Slide 14: Without `h-100`

If `h-100` is removed: - Cards take content height only - Buttons misalign - Layout looks unprofessional

---

Slide 15: Golden Rules (Very Important)

1. Grid total = 12
2. Bootstrap is mobile-first
3. Bigger screens override smaller
4. `.row` is flexbox
5. `h-100` = equal height cards

Slide 16: Common Student Mistakes

- Forgetting `.row`
- Mixing custom CSS before understanding grid
- Using `col-md-*` only and expecting mobile layout
- Not calculating 12 properly

Slide 17: Class Practice Task

Task: - Create 6 cards - Mobile → 1 card per row - Tablet → 2 cards per row - Desktop → 3 cards per row

Only change column classes

Slide 18: Final Summary

- Bootstrap Grid = layout backbone
- Cards + Grid = real websites
- Mastering grid = mastering responsive design

  If grid is clear, Bootstrap becomes easy