Sejauh ini kita telah mengembangkan aplikasi Wisata Bandung untuk platform mobile. Dengan Flutter mengumumkan Flutter Web yang sudah masuk versi stable, artinya terdapat target platform tambahan yang perlu kita kerjakan jika ingin menyasar pengguna yang lebih luas. Bukan hanya web, pada codelab kali ini kita juga akan menerapkan layout yang responsif untuk platform mobile untuk ukuran layar yang lebih besar seperti tablet.

Hasil akhir dari codelab ini akan seperti berikut:

Mari kita mulai!

1. Silakan buka project aplikasi Wisata Bandung Anda.
2. Pastikan di dalam proyek terdapat folder **web**. Apabila tidak ada, jalankan perintah berikut untuk mengaktifkan konfigurasi web.
   * flutter config --enable-web
3. Kemudian, untuk menambahkan target dukungan untuk web, jalankan perintah di bawah pada terminal:
   * flutter create .
4. Folder **web** akan terbangkitkan secara otomatis. Folder tersebut berisikan berkas pendukung web seperti index.html, manifest.json, dan lainnya.
5. Sekarang jalankan aplikasi Anda. Aplikasi akan berjalan dengan baik di browser. Akan tetapi kita akan menemui masalah dalam aspek ukuran, seperti list yang terlalu besar seperti berikut:
6. Karena kita telah membuat aplikasi mobile lalu ingin membuat versi webnya, maka secara tidak langsung kita telah menerapkan *mobile-first design*. Sekarang kita perlu menentukan pada ukuran berapa *layout mobile* ini sudah tidak sesuai dan perlu diubah tampilannya. Untuk memudahkan, mari kita tampilkan lebar browser atau layar ke dalam teks AppBar.
   * AppBar(
   * title:
   * Text('Wisata Bandung. Size: ${MediaQuery.of(context).size.width}'),
   * ),
7. Cobalah mengubah ukuran window browser dan cari breakpoint atau titik di mana layout sudah tidak sesuai.
8. Sebagai contoh, pada ukuran lebar di atas 600 sudah banyak ruang kosong yang lebih baik ditempati oleh widget lain. Maka, kita akan gunakan ukuran lebar 600 sebagai batas breakpoint ukuran mobile.
9. Pada ukuran lebar di atas 600 kita akan buat tampilan yang berbeda menggunakan grid. Sebelumnya, mari pindahkan tampilan **ListView** menjadi widget tersendiri.
   * class MainScreen extends StatelessWidget {
   * const MainScreen({Key? key}) : super(key: key);
   * @override
   * Widget build(BuildContext context) {
   * return Scaffold(
   * appBar: AppBar(
   * title:
   * Text('Wisata Bandung. Size: ${MediaQuery.of(context).size.width}'),
   * ),
   * **body: TourismPlaceList(),**
   * );
   * }
   * }
   * **class TourismPlaceList extends StatelessWidget {**
   * **const TourismPlaceList({Key? key}) : super(key: key);**
   * **@override**
   * **Widget build(BuildContext context) {**
   * **return ListView.builder(...);**
   * **}**
   * **}**
10. Kemudian buat widget baru untuk menampilkan **GridView**.
    * class TourismPlaceGrid extends StatelessWidget {
    * const TourismPlaceGrid({Key? key}) : super(key: key);
    * @override
    * Widget build(BuildContext context) {
    * return Padding(
    * padding: const EdgeInsets.all(24.0),
    * child: GridView.count(
    * crossAxisCount: 4,
    * children: [],
    * ),
    * );
    * }
    * }
11. Selanjutnya ubah body dari **MainScreen** untuk menampilkan **TourismPlaceList** atau **TourismPlaceGrid**. Kita akan memanfaatkan widget LayoutBuilderuntuk mendapatkan ukuran layar.
    * class MainScreen extends StatelessWidget {
    * @override
    * Widget build(BuildContext context) {
    * return Scaffold(
    * appBar: AppBar(
    * title:
    * Text('Wisata Bandung. Size: ${MediaQuery.of(context).size.width}'),
    * ),
    * **body: LayoutBuilder(**
    * **builder: (BuildContext context, BoxConstraints constraints) {**
    * **if (constraints.maxWidth <= 600) {**
    * **return TourismPlaceList();**
    * **} else {**
    * **return TourismPlaceGrid();**
    * **}**
    * **},**
    * **),**
    * );
    * }
    * }
12. Tampilan aplikasi di atas lebar 600 akan kosong karena kita belum mendefinisikan item untuk ditampilkan.
13. Tampilan item grid yang akan kita buat akan seperti di bawah ini. Cobalah untuk membuat tampilan Card seperti ini dulu sebelum melihat kode pada langkah berikutnya.
14. Berikut ini adalah tampilan untuk Card di atas:
    * GridView.count(
    * crossAxisCount: 4,
    * children: tourismPlaceList.map((place) {
    * return InkWell(
    * onTap: () {
    * Navigator.push(context, MaterialPageRoute(builder: (context) {
    * return DetailScreen(place: place);
    * }));
    * },
    * child: Card(
    * child: Column(
    * crossAxisAlignment: CrossAxisAlignment.stretch,
    * children: [
    * Expanded(
    * child: Image.asset(
    * place.imageAsset,
    * fit: BoxFit.cover,
    * ),
    * ),
    * const SizedBox(height: 8),
    * Padding(
    * padding: const EdgeInsets.only(left: 8.0),
    * child: Text(
    * place.name,
    * style: const TextStyle(
    * fontSize: 16.0,
    * fontWeight: FontWeight.bold,
    * ),
    * ),
    * ),
    * Padding(
    * padding: const EdgeInsets.only(left: 8.0, bottom: 8.0),
    * child: Text(
    * place.location,
    * ),
    * ),
    * ],
    * ),
    * ),
    * );
    * }).toList(),
    * ),
15. Jalankan aplikasi untuk melihat perubahan. Anda dapat melakukan duplikasi data dummy agar tampilan data menjadi semakin banyak.
16. Namun, untuk ukuran resolusi layar yang lebih besar lagi tampilan grid dengan empat kolom akan terlihat terlalu besar.  
    Geser kembali ukuran jendela browser untuk menemukan titik di mana tampilan sudah harus berubah. Misalnya, kita ambil ukuran lebar di atas 1200 akan menampilkan 6 kolom.
17. Tambahkan lagi kondisi pada **LayoutBuilder**. Kita akan menentukan jumlah kolom yang digunakan melalui parameter *constructor*.
    * LayoutBuilder(
    * builder: (BuildContext context, BoxConstraints constraints) {
    * if (constraints.maxWidth <= 600) {
    * return TourismPlaceList();
    * } else if (constraints.maxWidth <= 1200) {
    * return TourismPlaceGrid(gridCount: 4);
    * } else {
    * return TourismPlaceGrid(gridCount: 6);
    * }
    * },
    * ),
18. Tambahkan constructor pada kelas TourismPlaceGrid.
    * class TourismPlaceGrid extends StatelessWidget {
    * **final int gridCount;**
    * const TourismPlaceGrid({Key? key, **required this.gridCount**}) : super(key: key);
    * @override
    * Widget build(BuildContext context) {
    * return Padding(
    * padding: const EdgeInsets.all(16.0),
    * child: GridView.count(
    * **crossAxisCount: gridCount,**
    * children: tourismPlaceList.map((place) {
    * return InkWell(...);
    * }).toList(),
    * ),
    * );
    * }
    * }
19. Jalankan aplikasi dan lihat perubahan. Tampilan untuk lebar resolusi 1920 sudah lebih baik dari sebelumnya.
20. Selanjutnya mari tambahkan jarak antara item supaya tidak terlalu rapat.
    * GridView.count(
    * crossAxisCount: gridCount,
    * **crossAxisSpacing: 16,**
    * **mainAxisSpacing: 16,**
    * children: tourismPlaceList.map((place) {
    * return InkWell(...);
    * }).toList(),
    * ),
21. Jangan lupa untuk menghapus kembali teks ukuran pada AppBar.
    * AppBar(
    * title: Text('Wisata Bandung'),
    * ),
22. Selanjutnya kita akan mulai memodifikasi tampilan halaman detail. Seperti yang Anda lihat ketika menjalankan aplikasi, halaman detail tampak kurang estetik untuk ukuran resolusi yang lebar. Untuk ukuran lebar di atas 800, kita akan membuat tampilannya seperti ini:
23. Kita gunakan **LayoutBuilder**untuk memisahkan tampilan berdasarkan breakpoint.
    * class DetailScreen extends StatelessWidget {
    * final TourismPlace place;
    * const DetailScreen({Key? key, required this.place}) : super(key: key);
    * @override
    * Widget build(BuildContext context) {
    * return LayoutBuilder(
    * builder: (BuildContext context, BoxConstraints constraints) {
    * if (constraints.maxWidth > 800) {
    * return DetailWebPage(place: place);
    * } else {
    * return DetailMobilePage(place: place);
    * }
    * },
    * );
    * }
    * }
24. Pindahkan widget yang sudah ada ke dalam kelas **DetailMobilePage**.
    * class DetailMobilePage extends StatelessWidget {
    * final TourismPlace place;
    * const DetailMobilePage({Key? key, required this.place}) : super(key: key);
    * @override
    * Widget build(BuildContext context) {
    * return Scaffold(...);
    * }
    * }
25. Buat juga kelas serupa untuk **DetailWebPage**.
26. Tidak ada perbedaan konten yang ditampilkan pada halaman detail, tetapi hanya susunannya saja yang berbeda. Dari gambar sebelumnya apakah Anda dapat menganalisis bagaimana layoutnya disusun? Jika belum, maka kurang lebihnya akan seperti berikut:
27. Mari kita mulai dengan Column pertama. Tambahkan Text untuk judul, lalu Row untuk struktur layout di bawahnya. Tambahkan jarak menggunakan widget SizedBox.
    * Scaffold(
    * body: Column(
    * crossAxisAlignment: CrossAxisAlignment.start,
    * children: [
    * const Text(
    * 'Wisata Bandung',
    * style: TextStyle(
    * fontFamily: 'Staatliches',
    * fontSize: 32,
    * ),
    * ),
    * const SizedBox(height: 32),
    * Row(
    * children: [],
    * ),
    * ],
    * ),
    * );
28. Widget Row akan berisi children seperti berikut:
    * Row(
    * crossAxisAlignment: CrossAxisAlignment.start,
    * children: [
    * Expanded(
    * child: Column(
    * children: [],
    * ),
    * ),
    * const SizedBox(width: 32),
    * Expanded(
    * child: Card(...),
    * ),
    * ],
    * ),
29. Widget Column pertama, kita gunakan untuk menampilkan gambar dari aset dan daftar gambar dari internet.
    * Column(
    * children: [
    * ClipRRect(
    * child: Image.asset(place.imageAsset),
    * borderRadius: BorderRadius.circular(10),
    * ),
    * const SizedBox(height: 16),
    * Container(
    * height: 150,
    * padding: const EdgeInsets.only(bottom: 16),
    * child: ListView(
    * scrollDirection: Axis.horizontal,
    * children: place.imageUrls.map((url) {
    * return Padding(
    * padding: const EdgeInsets.all(4.0),
    * child: ClipRRect(
    * borderRadius: BorderRadius.circular(10),
    * child: Image.network(url),
    * ),
    * );
    * }).toList(),
    * ),
    * ),
    * ],
    * ),
30. Sementara Card untuk detail informasi tempat wisata memiliki widget seperti berikut:
    * Card(
    * child: Container(
    * padding: const EdgeInsets.all(16),
    * child: Column(
    * mainAxisSize: MainAxisSize.max,
    * children: <Widget>[
    * Container(
    * child: Text(
    * place.name,
    * textAlign: TextAlign.center,
    * style: const TextStyle(
    * fontSize: 30.0,
    * fontFamily: 'Staatliches',
    * ),
    * ),
    * ),
    * Row(
    * mainAxisAlignment: MainAxisAlignment.spaceBetween,
    * children: [
    * Row(
    * children: <Widget>[
    * const Icon(Icons.calendar\_today),
    * const SizedBox(width: 8.0),
    * Text(
    * place.openDays,
    * style: informationTextStyle,
    * ),
    * ],
    * ),
    * const FavoriteButton(),
    * ],
    * ),
    * Row(
    * children: <Widget>[
    * const Icon(Icons.access\_time),
    * const SizedBox(width: 8.0),
    * Text(
    * place.openTime,
    * style: informationTextStyle,
    * ),
    * ],
    * ),
    * const SizedBox(height: 8.0),
    * Row(
    * children: <Widget>[
    * const Icon(Icons.monetization\_on),
    * const SizedBox(width: 8.0),
    * Text(
    * place.ticketPrice,
    * style: informationTextStyle,
    * ),
    * ],
    * ),
    * Container(
    * padding: const EdgeInsets.symmetric(vertical: 16.0),
    * child: Text(
    * place.description,
    * textAlign: TextAlign.justify,
    * style: const TextStyle(
    * fontSize: 16.0,
    * fontFamily: 'Oxygen',
    * ),
    * ),
    * ),
    * ],
    * ),
    * ),
    * ),
31. Sekarang tampilan aplikasi akan seperti ini:
32. Mari tambahkan Padding untuk memberikan jarak antara konten dengan pinggir halaman.
    * Scaffold(
    * body: **Padding(**
    * **padding: const EdgeInsets.symmetric(**
    * **vertical: 16,**
    * **horizontal: 64,**
    * ),
    * child: Column(...),
    * ),
    * );
33. Kita juga dapat menentukan ukuran lebar maksimum dari konten yang tampil menggunakan SizedBox.
    * Padding(
    * padding: const EdgeInsets.symmetric(
    * vertical: 16,
    * horizontal: 64,
    * ),
    * **child: Center(**
    * **child: SizedBox(**
    * **width: 1200,**
    * child: Column(...),
    * ),
    * ),
    * ),
34. Lihat perubahan aplikasi, sekarang halaman detail akan memiliki ukuran lebar maksimal 1200 pixel.
35. Satu hal yang penting untuk kita perhatikan adalah menghadirkan pengalaman pengguna yang nyaman saat menggunakan platform apa pun. Untuk itu, tambahkan Scrollbar pada daftar gambar. Selain itu, tambahkan juga widget Container untuk memberikan ukuran tinggi pada ListView.
    * **Scrollbar(**
    * **child: Container(**
    * **height: 150,**
    * **padding: const EdgeInsets.only(bottom: 16),**
    * child: ListView(
    * scrollDirection: Axis.horizontal,
    * children: widget.place.imageUrls.map((url) {
    * return Padding(...);
    * }).toList(),
    * ),
    * ),
    * ),
36. Jika Anda menggunakan touchpad pada laptop, gestur swipe pada scrollbar akan berjalan dengan normal. Namun, ketika melakukan drag menggunakan mouse, fitur scrollbar tidak berjalan. Solusi yang bisa Anda lakukan adalah menggunakan ScrollController. Inisialisasikan objek ScrollController di atas method build().
    * final \_scrollController = ScrollController();
37. Tambahkan \_scrollController pada Scrollbar dan ListView.
    * Scrollbar(
    * **controller: \_scrollController,**
    * child: Container(
    * height: 150,
    * padding: const EdgeInsets.only(bottom: 16),
    * child: ListView(
    * **controller: \_scrollController,**
    * scrollDirection: Axis.horizontal,
    * children: widget.place.imageUrls.map((url) {
    * return Padding(...);
    * }).toList(),
    * ),
    * ),
    * ),
38. Seperti halnya TextEditingController, ScrollController juga harus di-dispose ketika widget sudah tidak lagi digunakan. Ubahlah DetailWebPage menjadi StatefulWidget supaya kita bisa memanggil method dispose.
39. Tambahkan method dispose setelah method build() lalu panggil \_scrollController.dispose();
    * @override
    * void dispose() {
    * \_scrollController.dispose();
    * super.dispose();
    * }
40. Selamat! Sekarang tampilan aplikasi Wisata Bandung sudah lebih sesuai untuk ukuran resolusi layar yang lebih besar. Anda telah belajar banyak menyusun halaman menggunakan widget, hingga membuat halaman yang dapat responsif terhadap perbedaan ukuran layar.

Kode selengkapnya untuk codelab ini dapat Anda temukan pada tautan:

* [Codelab-4](https://github.com/dicodingacademy/a159-flutter-pemula-labs/tree/main/wisatabandung/codelab-4)