Got it — you want to use `lib-ng-swiper` (Angular wrapper for Swiper) and **dynamically generate items at runtime** when the user navigates to the next slide.

This is a common **lazy-loading / infinite scroll** pattern for carousels.
The idea is:

- **Listen to Swiper events** (`slideChange`, `reachEnd`, etc.).
- **Append new items** to your data array when the event occurs.
- **Trigger Swiper to update** so it recognizes the new slides.

Example: Angular 17 + lib-ng-swiper

```
// app.component.ts
import { Component } from '@angular/core';
import { SwiperOptions } from 'swiper/types';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  slides: string[] = [];
  slideCount = 0;

  swiperConfig: SwiperOptions = {
    slidesPerView: 1,
    spaceBetween: 10,
    navigation: true,
    pagination: { clickable: true },
    on: {
      slideChange: (swiper) => this.onSlideChange(swiper),
      reachEnd: () => this.loadMoreSlides()
    }
  };

  constructor() {
    // Initial load
    this.loadMoreSlides();
  }

  // Called when slide changes
  onSlideChange(swiper: any) {
    const currentIndex = swiper.activeIndex;
    // Example: load more when user is 2 slides from the end
    if (currentIndex >= this.slides.length - 2) {
      this.loadMoreSlides();
    }
  }

  // Append new slides
  loadMoreSlides() {
    for (let i = 0; i < 5; i++) {
      this.slideCount++;
      this.slides.push(`Slide #${this.slideCount}`);
    }
  }
}
```

```
<!-- app.component.html -->
<swiper-container [config]="swiperConfig" style="width: 100%; height:
300px;">
  <swiper-slide *ngFor="let slide of slides">
    <div class="slide-content">
      {{ slide }}
    </div>
  </swiper-slide>
</swiper-container>
```

```
/* app.component.css */
.slide-content {
  display: flex;
  align-items: center;
  justify-content: center;
  background: #1976d2;
  color: white;
  font-size: 1.5rem;
  height: 100%;
}
```

Key Points

- `reachEnd` event is perfect for infinite scroll.
- `slideChange` can be used if you want to preload before reaching the end.
- No need to manually call `swiper.update()` in Angular — the wrapper detects array changes automatically if using `*ngFor`.
- If you use **virtual slides** for performance, you'll need to call `swiper.virtual.update()` after pushing new items.

✅ This approach **generates new slides at runtime** when the user navigates forward, without preloading everything at once.

If you want, I can give you a **version with virtual slides** so it can handle **thousands of items without performance issues**.
Do you want me to prepare that?