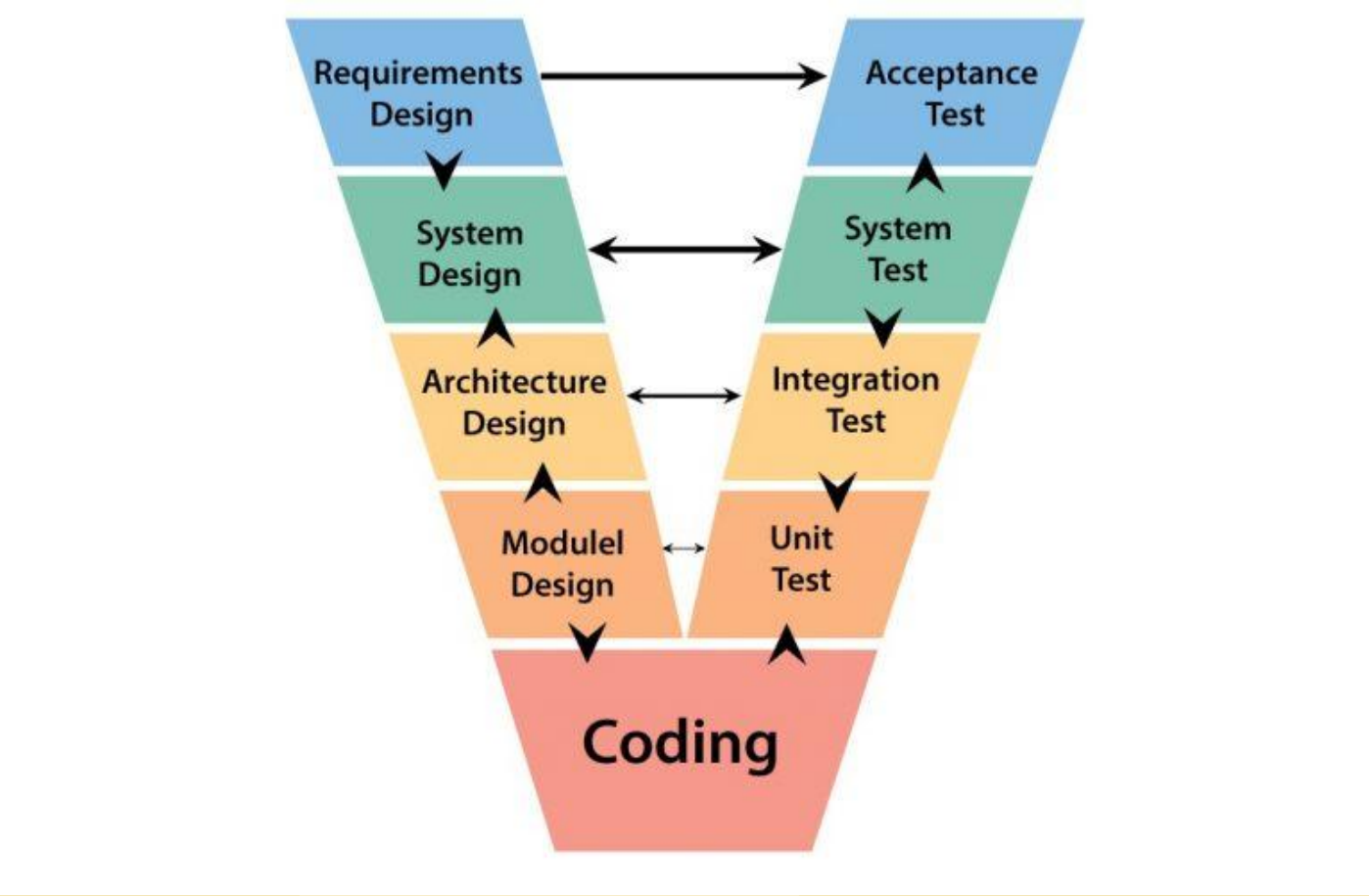
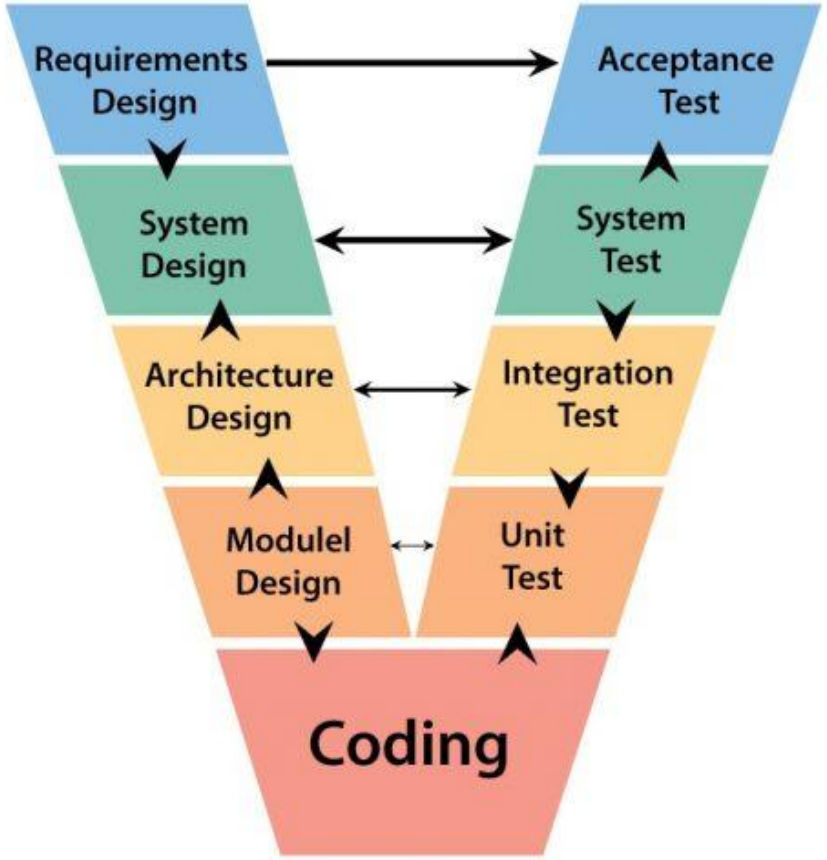
ERHVERVSAKADEMI KØBENHAVN

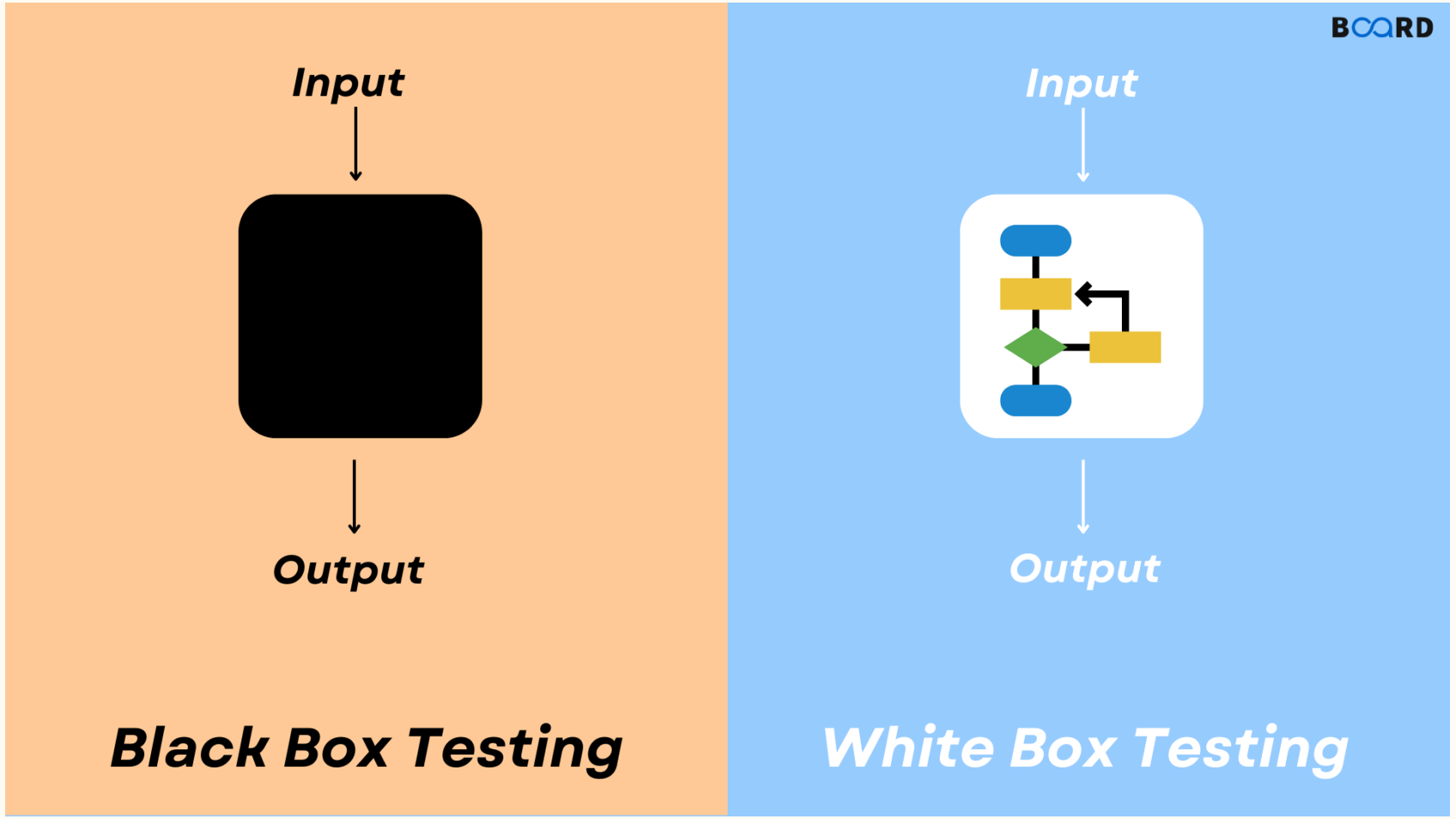
Test



Der er mange slags test



White box og black box



Strukturerede test

Kriterium	Strukturerede tests	Uformelle tests ("klikke rundt")
Genskabelighed	✅ Kan gentages præcist med samme input og forventet output.	❌ Kan være svært at genskabe nøjagtigt.
Dokumentation	✅ Test cases og resultater er dokumenteret.	❌ Ingen eller minimal dokumentation.
Dækning af edge cases	✅ Grænsetilfælde testes systematisk.	⚠️ Kun hvis testeren tilfældigt støder på dem.
Reproducerbare fejl	✅ Fejl kan let genskabes og rapporteres.	❌ Fejl kan være svære at gentage.
Uafhængighed	✅ Testcases er ofte gensidigt uafhængige, så fejl ikke skjules.	❌ Tests afhænger af, hvor testeren klikker.
Automatisering	✅ Kan ofte automatiseres.	❌ Meget afhængig af manuelt arbejde.
Foruddefinerede kriterier	✅ Har klare kriterier for succes/fiasko.	❌ Ingen faste kriterier – testeren vurderer selv.
Skalérbarhed	✅ Kan nemt udvides til større testscenarier.	❌ Svært at skalere og reproducere.

Hvordan vælger man testcases?

- Sunny day – input er som forventet
- Grænsetilfælde, fx:

Kategori	Eksempel på input	Forventet udfald
Numerisk grænse	Minimum og maksimum værdi, f.eks. alder: `0` og `120`.	Systemet håndterer begge korrekt.
Tomt input	En tom tekstboks eller en liste uden elementer.	Ingen fejl, men passende håndtering (fx en fejlmeddelelse eller default-værdi).
Maksimal længde	Bruger skriver et navn med `255` tegn (hvis feltet har en grænse på `255`).	Ingen fejl, og data håndteres korrekt.
Specielle tegn	Indsæt `'; DROP TABLE users;` i et inputfelt.	Systemet beskytter mod SQL-injektion.
Grænse mellem to tilstande	Rabat gives ved køb over `500 kr.` – test med `499,99` og `500,00`.	Rabatten gives kun korrekt ved `500,00` eller mere.
Tidspunkt på kanten	Test nytårsaften `23:59:59` eller skift til sommertid.	Systemet håndterer datoændringer korrekt.

A software tester walks into a bar.

Runs into a bar.

Crawls into a bar.

Dances into a bar.

Flies into a bar.

Jumps into a bar.

And orders:

a beer.

2 beers.

0 beers.

99999999 beers.

a lizard in a beer glass.

-1 beer.

“qwertyuiop” beers.

Testing complete.

A real customer walks into the bar and asks where the bathroom is.

The bar goes up in flames.

Hvornår laver vi integrationstest?

- Programmet kommunikerer med en database
- Programmet kommunikerer over netværk
- Programmet rører filsystemet
- Du har brug for at gøre specielle ting ved miljøet for at køre testen

Test og dependency injection

Test-base

```java
public class BookMapperTest {

    private static final String USER = "postgres";
    private static final String PASSWORD = "postgres";
    private static final String URL =
            "jdbc:postgresql://localhost:5432/library?currentSchema=test";

    private static ConnectionPool connectionPool;
    private static BookMapper bookMapper;

    @BeforeAll
    static void setUpClass() {
        connectionPool =
                ConnectionPool.getInstance(USER, PASSWORD, URL, db: "");

        bookMapper = new BookMapper(connectionPool);
```

Prod-base

```java
public class Main {

    private static final String USER = "postgres";
    private static final String PASSWORD = "postgres";
    private static final String URL =
            "jdbc:postgresql://localhost:5432/%s?currentSchema=public";
    private static final String DB = "library";

    private static final ConnectionPool connectionPool =
            ConnectionPool.getInstance(USER, PASSWORD, URL, DB);


    public static void main(String[] args){

        var app = Javalin.create( JavalinConfig config -> {
            config.staticFiles.add("/public");
            config.fileRenderer
                    (new JavalinThymeleaf(ThymeleafConfig.templateEngine()));
            UserController userController = new UserController(connectionPool);
            CartController cartController = new CartController(connectionPool);
            BookController bookController = new BookController(connectionPool);
```

Brug af testdatabase

1. Opret et schema ”test” i din database
2. I ”test” oprettes de samme tabeller som i ”public” dog uden data.
3. Gør det muligt for mapper’en at bruge en anden forbindelse (vha. dependency injection).
4. Før hver test fjernes alle rækker i testdatabasens tabeller, så der startes på en frisk.
5. Når alle rækker er fjernet, indsættes et antal rækker i tabellerne, så man har fuldkommen styr på hvilke data, som ligger i testdatabasen før hver test køres.
6. Skriv test metoder
7. Kør dine test