# Test Driven Hotel

| | |
|---|---|
| Typ | Labb |
| Ämnen | CI/CD, Razor Pages, TDD, Unit Testing |

🎯 Utveckla en del av ett Hotellbokningssystem med TDD och alternativt CI/CD-pipeline.

## Uppgift

### 1. Förstå och definiera kraven

- **Mål:** Definiera funktioner för ditt system.
- **Aktiviteter:**
  - Formulera användarberättelse(r) (minst 1!)
  - Lista funktioner.

💡 **Exempel på Användarberättelse**

- **Berättelse:** Som en hotellgäst vill jag kunna söka och boka tillgängliga rum via en webbsida.
- **Acceptanskriterier:**
  - Möjlighet att välja in- och utcheckningsdatum och rumstyp (enkel-, dubbelrum) på en Razor Page-sida.
  - Visa en lista över tillgängliga rum som matchar kriterierna.
  - Möjlighet att boka ett rum genom webbgränssnittet.

### 2. Designa systemarkitekturen

- **Mål:** Skapa en plan för ditt system.
- **Aktiviteter:**
  - Utveckla en högnivåarkitektur, affärslogik och Razor Pages-gränssnitt (minst 2 lager).
  - Definiera datastrukturer och modeller.

### 3. Skriv testfall med xUnit och FluentAssertions

- **Mål:** Skriva testfall för varje funktion och aspekt av systemet.
- **Aktiviteter:**
  - Skapa ett testprojekt och skriv testfall för varje funktion du definerat ovan (dina **Användarberättelser)**.
  - Fundera på vilka “edge cases” varje funktion har, och skriv test för dessa.

### 4. Implementera funktionalitet med TDD

- **Mål:** Utveckla systemet enligt TDD-processen (Red-Green-Refactor).
- **Aktiviteter:**

- Skriv kod för att få testen att passera.
- Refaktorera koden för förbättringar.

### 5. Skapa enkelt användargränssnitt

- **Mål:** Skapa ett enkelt användargränssnitt med ASP.NET Core Razor Pages.
- **Aktiviteter:**
  - Använd Razor Pages för att bygga gränssnittet för ditt system.
  - Integrera gränssnittet med bakomliggande affärslogik för din(a) användarberättelse(r).

### 6. VG: CI/CD

- Mål: Introducera kontinuerlig integration, där dina testkörningar och kodincheckningar hanteras automatiskt via GitHub Actions.
- **Aktiviteter:**
  - Skapa en CI/CD pipeline som kör alla tester när en push till main branchen görs.
  - Om testerna går igenom, publicera projektet till Azure som en web app.

## Inlämningskrav för Hotellbokningssystemprojektet

### 1. Kodbas och repository

- Ladda upp hela projektet till ett GitHub-repository. Se till att inkludera alla nödvändiga filer så att projektet kan klonas och köras utan ytterligare konfiguration.
- Repositoryt ska innehålla en README-fil där din användarberättelse tydligt kan utläsas.

### 2. Kodkvalitet och Organisation

- Se till att koden är väl organiserad, ren och lätt att läsa. Använd tydliga namn för variabler, metoder och klasser.
- Inkludera kommentarer där det är nödvändigt för att förklara komplex logik eller viktiga beslut.

### 3. Testfall

- Se till att du har skrivit enhetstester för din affärslogik. Testerna ska täcka viktiga scenarier och “edge cases”.
- Inkludera en rapport eller screenshots som visar resultatet av dina testkörningar. Inkludera dessa screenshots i repositoryt.

### 4. Funktionalitet

- Projektet ska implementera minst 1 användarberättelse med tillhörande funktioner som definierats i uppgiften.
- Razor Pages-sidorna bör vara användarvänliga och tillhandahålla enkel navigering och interaktion med systemet.

### 5. VG: Reflektion och utvärdering

- Skriv en kort reflektion om projektet. Beskriv vad du lärde dig, vilka utmaningar du stötte på, och hur du övervann dem. Om du gjorde om projektet när det var klart, vad hade du gjort annorlunda då?

## Inlämningsinstruktioner

- **Deadline:** Se till att lämna in projektet före utsatt deadline (11/2 2024 kl. 23:59)
- **Inlämningsmetod:** Lämna in länken till ditt GitHub-repository via Learnpoint.

## Noteringar

- Det är inte ett krav att implementera en databas i projektet, använd gärna istället en statisk lista för att hantera datan.
    - När appen publiceras om kommer den statiska listan att rensas, men det är okej!