

2024 - 2025 Bahar Dönemi

Yazılım ve Kalite Güvence Temelleri Dersi Proje Raporu

**Dersten Sorumlu Öğretmen:** Hilal RAKICI

**Öğrenci Adı Soyadı / Numarası:** Ceyhun AY - 170420844

Umut Eren TORAMAN - 170421037

Özgür UYLAŞ - 171421008

Çağan DERBENT - 170421028

**Proje Konusu:** Bu proje kapsamında, seçilen “Automation In Testing” uygulamasının API’si test edilmiş ve aynı uygulamanın kullanıcı arayüzü (UI) Selenium ile test edilmiştir.

**Rapor İçeriği**

**1. API Testleri**

  1.1 Uygulama ve API Analizi  
  1.2 Oluşturulan Test Senaryoları  
  1.3 Test Sonuçları (Başarılı ve Başarısız Senaryolar)  
  1.4 Karşılaşılan Sorunlar ve Çözümleri

**2. UI Testleri**

  2.1 Test1: Sayfa Başlığı Kontrolü  
  2.2 Test 2: Geçersiz Telefon Numarası ile Form Hata Kontrolü  
  2.3 Test 3: E-posta Boş Bırakıldığında Uyarı  
  2.4 Test 4: Çok Uzun Mesaj Gönderildiğinde Uyarı

2.5 Test 5: Admin Paneli Giriş Formu Görünürlük Testi

2.6 Test 6: Yanlış Admin Giriş Bilgileri ile Giriş Denemesi

2.7 Test 7: Geçerli Admin Girişi ile Panel Erişimi

2.8 Test 8: Formun Tamamen Boş Gönderilmesi Durumunda Hata Mesajı Kontrolü

2.9 Test 9: Rooms ve Contact Linkleri Fonksiyonellik Testi

2.10 Test 10: Mobil Menü Fonksiyonellik Testi

2.11 Test 11: 404 Sayfası Görünürlük Testi

2.12 Test 12: Oda Mevcutluğu ve Yönlendirme Testi

**1. API Testleri**

* 1. **Uygulama ve API Analizi**

Proje kapsamında test edilen uygulama, https://restful-booker.herokuapp.com adresinde çalışan bir rezervasyon yönetim sistemidir. RESTful mimariye sahip bu sistem; rezervasyon oluşturma, görüntüleme, güncelleme ve silme işlemlerini desteklemektedir.

**Test Edilen Endpoint’ler**

|  |  |  |
| --- | --- | --- |
| **Endpoint** | **Metod** | **Amaç** |
| /auth | POST | Giriş yaparak token almak |
| /booking | POST | Yeni rezervasyon oluşturmak |
| /booking/{id} | GET | Rezervasyon görüntülemek |
| /booking/{id} | PUT | Rezervasyon güncellemek |
| /booking/{id} | DELETE | Rezervasyon silmek |

* 1. **Oluşturulan Test Senaryoları**

Proje sürecinde hem başarılı hem de hatalı senaryolar tanımlanmış ve test edilmiştir.  
Senaryolarda test script’leri JavaScript kullanılarak yazılmış ve Postman ortam değişkenleri (token, bookingId) ile otomasyon sağlanmıştır.

**Başarılı Senaryolar**

|  |  |
| --- | --- |
| **Test Numarası** | **Açıklama** |
| 01 | Doğru bilgilerle giriş → token alma |
| 03 | Geçerli body ile rezervasyon oluşturma |
| 05 | Geçerli ID ile rezervasyon görüntüleme |
| 07 | Geçerli token ile rezervasyon güncelleme |
| 09 | Geçerli token ile rezervasyon silme |

**Başarısız Senaryolar**

|  |  |
| --- | --- |
| **Test Numarası** | **Açıklama** |
| 02 | Yanlış şifre ile giriş (token alınamamalı) |
| 04 | Eksik body ile rezervasyon denemesi |
| 06 | Geçersiz ID ile rezervasyon sorgulama |
| 08 | Token olmadan rezervasyon güncelleme |
| 10 | Token olmadan rezervasyon silme |

**Örnek Senaryolar**

* **03 - Rezervasyon Oluşturma (Başarılı)**

**HTTP Metodu:** POST **Endpoint:** /booking

**Amaç:** Geçerli verilerle yeni bir rezervasyon oluşturulup oluşturulamadığını test etmek

**Request Header:** Content-Type: application/json

Accept: application/json

**Request Body:** {

"firstname": "Ceyhun",

"lastname": "Ay",

"totalprice": 150,

"depositpaid": true,

"bookingdates": {

"checkin": "2025-06-01",

"checkout": "2025-06-10"

},

"additionalneeds": "Breakfast"

}

**Beklenen Sonuç:** HTTP Status: 200 OK

Response body'de bookingid ve gönderilen bilgiler dönmeli

**Script (Postman / JavaScript Test):** pm.test("Status code 200 veya 201 olmalı", function () {

pm.expect(pm.response.code).to.be.oneOf([200, 201]);

});

pm.test("Rezervasyon ID dönmeli", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.bookingid).to.not.be.undefined;

pm.environment.**set**("bookingId", jsonData.bookingid);

});

pm.test("İsim doğru dönmeli", function () {

  var jsonData = pm.response.json();

    pm.expect(jsonData.booking.firstname).to.eql("Ceyhun");

});

pm.test("Yanıt süresi 1000ms'den kısa olmalı", function () {

    pm.expect(pm.response.responseTime).to.be.below(1000);

});

* **08 – Rezervasyon Güncelleme (Yetkisiz)**

**HTTP Metodu:** PUT **Endpoint:** /booking/{{bookingId}}

**Amaç:** Kullanıcı doğrulama (authentication) yapılmadan rezervasyon güncelleme girişiminin engellenip engellenmediğini test etmek

**Request Header:** Content-Type: application/json

Accept: application/json

**Request Body:** {

"firstname": "Yetkisiz",

"lastname": "Güncelleme",

"totalprice": 999,

"depositpaid": true,

"bookingdates": {

"checkin": "2025-07-01",

"checkout": "2025-07-10"

},

"additionalneeds": "Lunch"

}

**Beklenen Sonuç:** HTTP Status: 403 Forbidden

Sunucu, token olmadığından güncellemeye izin vermemelidir

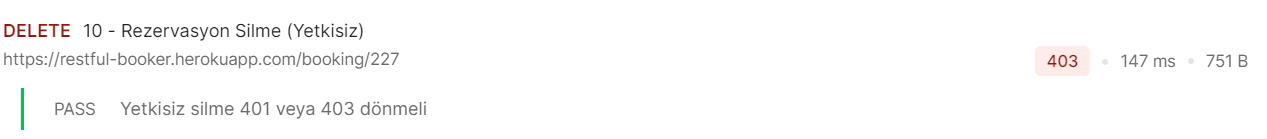
**Script (Postman / JavaScript Test):** pm.test("403 veya 401 dönmeli", function () {

pm.expect(pm.response.code).to.be.oneOf([401, 403]);

});

* 1. **Test Sonuçları (Başarılı ve Başarısız Senaryolar)**

Testler Postman Collection Runner ile tek seferde çalıştırılmıştır. Tüm senaryolar başarıyla yürütülmüş; sistemin doğruluk, hata kontrolü ve performansı test edilmiştir.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

* 1. **Karşılaşılan Sorunlar ve Çözümleri**

**Sorun 1**

Login testi ilk denemede 3609 ms ile başarısız oldu. Bu durum yoğunluk kaynaklı olup, sonraki çalıştırmada 659 ms’ye düştü ve başarıyla tamamlandı. Tüm başarılı testlerde aşağıdaki script kullanılarak performans ölçüldü:

pm.test("Yanıt süresi 1000ms'den kısa olmalı", function () {

pm.expect(pm.response.responseTime).to.be.below(1000);

});

**Sorun 2**

Bazı test senaryolarında, RESTful API tasarımı gereği sistem beklenenden farklı HTTP durum kodları döndürmüştür. Bu durumlar testlerde özel olarak ele alınmış, yanıt gövdesi (body) üzerinden doğrulama yapılmıştır.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No** | **Açıklama** | **Beklenen Durum** | **Gerçek Yanıt** | **Alınan Aksiyon** |
| 02 | Hatalı kullanıcı adı/şifre ile giriş | 401 / 403 | 200 OK + "Bad credentials" | Yanıt body içeriği test edilerek kontrol sağlandı |
| 04 | Eksik body ile rezervasyon oluşturma | 400 / 422 | 500 Internal Server Error | Yanıt düz metin içerdiği için pm.response.text() ile test edildi |

**2. UI Testleri**

Bu bölümde, uygulamanın kullanıcı arayüzüne yönelik olarak gerçekleştirilen otomasyon testlerine yer verilmiştir. API testlerinin ardından, sistemin uçtan uca doğrulanabilmesi amacıyla arayüz üzerinden gerçekleştirilen kullanıcı senaryoları da test kapsamına dahil edilmiştir.

UI testleri, Python dili ile geliştirilmiş olup Selenium kütüphanesi kullanılarak yürütülmüştür. Testler, kullanıcı davranışlarını gerçekçi bir şekilde simüle ederek sayfa geçişleri, form girişleri, buton etkileşimleri ve beklenen doğrulama mesajları gibi unsurların kontrolünü sağlamaktadır. Amaç, son kullanıcının karşılaşabileceği arayüz akışlarının doğru ve kararlı bir şekilde çalıştığının garanti altına alınmasıdır.

Her bir test senaryosu, belirli bir işlevselliği doğrulamak amacıyla kurgulanmış olup, test adımları, beklenen sonuçlar ve testin durumu (başarılı/başarısız) raporlanmaktadır. Bu testler sayesinde hem işlevsel hataların hem de görsel uyumsuzlukların erken aşamada tespit edilmesi hedeflenmiştir.

**2.1 Test 1: Sayfa Başlığı Kontrolü**

**Amaç:**  
Uygulama ana sayfasına erişildiğinde, tarayıcı sekmesinde görünen başlık bilgisinin beklendiği gibi olup olmadığını kontrol etmektir.

**Senaryo:**  
Kullanıcı, https://automationintesting.online/ adresine eriştiğinde sayfa başlığı "Restful-booker-platform demo" olmalıdır. Bu kontrol, uygulamanın doğru sayfaya yönlendirdiğini ve sayfanın başarıyla yüklendiğini doğrulamak amacıyla yapılmaktadır.

**Adımlar:**

1. Chrome tarayıcı başlatılır.
2. Belirtilen URL'ye gidilir.
3. Sayfa başlığı (driver.title) alınır.
4. Beklenen başlık ile karşılaştırma yapılır.

**Beklenen Sonuç:**

Sayfa başlığı "Restful-booker-platform demo" ile tam olarak eşleşmelidir.

**Test Sonucu:**

Başlık eşleşiyorsa test geçer; aksi durumda test başarısız olur ve fark bilgisi raporlanır.

**2.2 Test 2: Geçersiz Telefon Numarası ile Form Hata Kontrolü**

**Amaç:**İletişim formuna geçersiz biçimde girilen telefon numarası karşısında sistemin doğru hata mesajı gösterip göstermediğini kontrol etmek.

**Senaryo:**Kullanıcı, iletişim formunu geçersiz bir telefon numarası (örneğin "abc123") girerek doldurur ve gönder butonuna tıkladığında, sistemin kullanıcıyı uygun bir uyarı mesajıyla bilgilendirmesi beklenir.

**Adımlar:**

1. Uygulama ana sayfası açılır.
2. Sayfa ortasına kaydırma yapılır (iletişim formunun görünebilmesi için).
3. Form alanları aşağıdaki şekilde doldurulur:

**İsim:** Ali

**E-posta:** ali@example.com

**Telefon:** abc123 *(geçersiz giriş)*

**Konu:** Geçersiz Telefon Testi

**Açıklama:** Bu test geçersiz telefon numarası içindir.

4.Submit (Gönder) butonuna tıklanır.

5.Sayfa üzerindeki hata mesajları incelenir.

**Beklenen Sonuç:**

Form gönderimi reddedilmeli ve "phone" (telefon) alanına ilişkin uyarı mesajı kullanıcıya gösterilmelidir.

**Test Sonucu:**

Eğer hata mesajları arasında telefonla ilgili bir uyarı varsa test geçer. Eğer sistem geçersiz numaraya rağmen uyarı vermezse test başarısız olur.

**2.3 Test 3: E-posta Boş Bırakıldığında Uyarı**

**Amaç:**İletişim formunda e-posta alanı boş bırakıldığında sistemin kullanıcıyı doğru bir şekilde uyarıp uyarmadığını kontrol etmek.

**Senaryo:**Kullanıcı, formu doldururken e-posta alanını boş bırakır. Form gönderildiğinde sistemin “E-posta boş bırakılamaz” şeklinde bir uyarı mesajı göstermesi beklenir.

**Adımlar:**

1. Ana sayfa açılır ve formun bulunduğu bölüme kaydırma yapılır.
2. Form alanları aşağıdaki şekilde doldurulur:

**İsim:** Özgür

**Telefon:** 55512345678

**Konu:** Zorunlu Alan Testi

**Açıklama:** E-posta boş bırakılacak.

**E-posta:** *Boş bırakılır*

1. Submit butonuna tıklanır.
2. Sayfadaki hata mesajları kontrol edilir.

**Beklenen Sonuç:**

E-posta alanı boş bırakıldığında “Email may not be blank” içerikli bir uyarı mesajı gösterilmelidir.

**Test Sonucu:**

Eğer belirtilen uyarı mesajı görünürse test başarılıdır. Uyarı mesajı yoksa veya farklı bir mesaj varsa test başarısız kabul edilir.

**2.4 Test 4: Çok Uzun Mesaj Gönderildiğinde Uyarı**

**Amaç:**İletişim formunda açıklama (description) alanına izin verilen karakter sınırının üzerinde giriş yapıldığında, sistemin uygun bir uyarı mesajı gösterip göstermediğini test etmek.

**Senaryo:**Kullanıcı, açıklama alanına 2000 karakterden uzun bir metin girer (örneğin 2100 karakter). Bu durumda sistemin kullanıcıyı “Mesaj 20 ile 2000 karakter arasında olmalıdır” benzeri bir uyarıyla bilgilendirmesi beklenir.

**Adımlar:**

1. Uygulama ana sayfası açılır.
2. Sayfa, form bölümünün görünmesi için aşağı kaydırılır.
3. Form alanları aşağıdaki gibi doldurulur:
   * İsim: Özgür
   * E-posta: ozgur@example.com
   * Telefon: 55512345678
   * Konu: Uzun Mesaj
   * Açıklama: 2100 karakterlik metin (örnek olarak "a" \* 2100)
4. Submit (Gönder) butonuna tıklanır.
5. Sayfada hata mesajı olup olmadığı kontrol edilir.

**Beklenen Sonuç:**

Sistem, açıklama alanına girilen metnin uzunluğu nedeniyle “Message must be between 20 and 2000 characters” şeklinde bir uyarı mesajı göstermelidir.

**Test Sonucu:**

Belirtilen uyarı görünürse test başarılıdır. Uyarı mesajı yoksa ya da farklıysa test başarısız kabul edilir.

**2.5 Test 5: Admin Paneli Giriş Formu Görünürlük Testi**

**Amaç:**  
Admin paneline erişim sağlandığında, kullanıcı adı ve şifre alanlarını içeren giriş formunun sayfada düzgün şekilde yüklenip yüklenmediğini kontrol etmek.

**Senaryo:**  
Kullanıcı doğrudan admin giriş sayfası URL’sine (/admin) eriştiğinde, giriş formu (kullanıcı adı ve şifre alanları) görünür olmalıdır. Bu formun eksik ya da yüklenmemiş olması, erişim sorunlarına veya güvenlik açıklarına neden olabilir.

**Adımlar:**

1. Tarayıcı açılır ve https://automationintesting.online/admin adresine gidilir.
2. Sayfa yüklendikten sonra <form> elementi beklenir.
3. Form içindeki aşağıdaki alanların varlığı kontrol edilir:

Kullanıcı adı giriş alanı (input[type='text'] veya input[name='username'])

Şifre giriş alanı (input[type='password'])

**Beklenen Sonuç:**

Giriş formu eksiksiz ve görünür olmalıdır. Her iki giriş alanı (kullanıcı adı ve şifre) sayfada yer almalıdır.

**Test Sonucu:**

Form öğeleri başarılı bir şekilde bulunursa test geçer. Formun yüklenmemesi veya alanlardan birinin eksik olması durumunda test başarısız olur ve hata mesajı detaylarıyla birlikte raporlanır.

**2.6 Test 6: Yanlış Admin Giriş Bilgileri ile Giriş Denemesi**

**Amaç:**  
Admin paneline hatalı kullanıcı adı ve şifre kombinasyonu ile giriş yapılmaya çalışıldığında, sistemin bu giriş denemesini reddedip uygun bir hata mesajı gösterip göstermediğini kontrol etmek.

**Senaryo:**  
Kullanıcı, admin giriş formuna yanlış bilgiler (örneğin "wronguser" ve "wrongpass") girerek oturum açmayı dener. Bu durumda sistemin güvenlik gereği kullanıcıyı reddetmesi ve kullanıcıya bir hata mesajı göstermesi beklenir.

**Adımlar:**

1. https://automationintesting.online/admin adresine gidilir.
2. Giriş formu alanları (kullanıcı adı ve şifre) sayfada yüklendiğinde:
   * **Kullanıcı Adı:** wronguser
   * **Şifre:** wrongpass
3. "Login" butonuna tıklanır.
4. Sayfada hata mesajı (alert-danger sınıfı) aranır.

**Beklenen Sonuç:**

Sistem, yanlış bilgilerle yapılan giriş denemesini reddetmeli ve kullanıcıya açık bir hata mesajı göstermelidir (örneğin: "Invalid username or password").

**Test Sonucu:**

Hata mesajı görünürse test geçer. Hatalı girişe rağmen sistem herhangi bir uyarı vermezse veya kullanıcıyı kabul ederse test başarısız olur.

**2.7 Test 7: Geçerli Admin Girişi ile Panel Erişimi**

**Amaç:**  
Doğru admin kullanıcı adı ve şifresi ile giriş yapıldığında, sistemin kullanıcıyı başarılı şekilde oturum açarak admin paneline yönlendirip yönlendirmediğini kontrol etmek.

**Senaryo:**  
Kullanıcı, sistemin tanımlı geçerli kimlik bilgileri olan "admin" ve "password" ile giriş yapar. Giriş başarılıysa kullanıcı admin paneline erişmeli ve panel bileşenleri (örneğin oda listesi, "Add Room", "Logout" vb.) görünmelidir.

**Adımlar:**

1. https://automationintesting.online/admin adresine gidilir.
2. Giriş formu alanları doldurulur:

**Kullanıcı Adı:** admin

**Şifre:** password

1. "Login" butonuna tıklanır.
2. Sayfa kaynak kodu kontrol edilerek aşağıdaki içeriklerin varlığı incelenir:

Rooms

Add Room

Logout

roomName

**Beklenen Sonuç:**

Giriş başarılı olmalı ve admin paneline ait içerikler sayfada yer almalıdır.

**Test Sonucu:**

Panel anahtar kelimeleri sayfa içinde bulunursa test başarılı sayılır. Eğer bu içerikler görünmezse, test başarısız kabul edilir ve erişim sorunları araştırılmalıdır.

**2.8 Test 8: Formun Tamamen Boş Gönderilmesi Durumunda Hata Mesajı Kontrolü**

**Amaç:**  
İletişim formunun hiçbir alanı doldurulmadan gönderilmesi durumunda, sistemin kullanıcıya eksik alanlara dair uygun uyarı mesajları gösterip göstermediğini test etmek.

**Senaryo:**  
Kullanıcı hiçbir alanı doldurmadan formu göndermeye çalışır. Bu durumda sistemin, eksik/zorunlu alanlara dair hata mesajları üretmesi ve kullanıcıyı bilgilendirmesi gerekir.

**Adımlar:**

1. Ana sayfa (https://automationintesting.online/) açılır.
2. Sayfa, form alanlarının görünmesi için aşağıya kaydırılır.
3. **İsim**, **E-posta**, **Telefon**, **Konu** ve **Açıklama** alanları boş bırakılır.
4. “Submit” butonuna tıklanır.
5. Sayfada oluşan hata mesajları alert-danger sınıfı üzerinden tespit edilir.

**Beklenen Sonuç:**

Sistem, eksik alanlar için aşağıdakilere benzer hata mesajlarını göstermelidir:

* "Name may not be blank"
* "Email may not be blank"
* "Phone may not be blank"
* "Subject may not be blank"
* "Description may not be blank"

**Test Sonucu:**

Hata mesajlarının tamamı sayfada görünüyorsa test başarılıdır. Eğer hata mesajı görünmezse ya da eksikse, test başarısız olarak değerlendirilir.

**2.9 Test 9: Rooms ve Contact Linkleri Fonksiyonellik Testi**

**Amaç:**  
Web sitesindeki "Rooms" ve "Contact" linklerinin doğru çalışıp çalışmadığını, ilgili sayfa ve formların sorunsuz şekilde yüklendiğini doğrulamak.

**Senaryo:**  
Kullanıcı ana sayfadan "Rooms" linkine tıklayarak odaların listelendiği sayfaya erişir. Ardından "Contact" linkine tıklayarak iletişim formunun eksiksiz göründüğünü kontrol eder.

**Adımlar:**

1. Test ortamında tarayıcı açılır ve https://automationintesting.online/ adresine gidilir.
2. Ana sayfadaki "Rooms" bağlantısı bulunur ve tıklanır.
3. "Rooms" sayfasının yüklendiği ve id="rooms" elementinin görünür olduğu doğrulanır.
4. Ana sayfada "Contact" bağlantısı bulunur, sayfa bu bağlantıya kaydırılır ve tıklanır.
5. İletişim formundaki zorunlu alanlar (name, email, phone, subject, description) görünürlük ve erişilebilirlik açısından kontrol edilir.

**Beklenen Sonuç:**

"Rooms" bağlantısına tıklanınca ilgili sayfa başarıyla yüklenir ve "Rooms" bölümü görünür olur. "Contact" bağlantısına tıklanınca iletişim formundaki tüm gerekli alanlar eksiksiz ve görünür şekilde kullanıcıya sunulur.

**Test Sonucu:**

"Rooms" bağlantısı ve bölümü başarıyla yüklendi."Contact" bağlantısı çalıştı ve iletişim formundaki tüm alanlar eksiksiz olarak göründü. Test başarılı şekilde tamamlandı.

**2.10 Test 10: Mobil Menü Fonksiyonellik Testi**

**Amaç:**  
Mobil görünümde hamburger menü simgesinin (navbar-toggler) düzgün çalıştığını ve menünün açılarak görünür olduğunu doğrulamak.

**Senaryo:**  
Kullanıcı mobil cihaz görünümünde siteye erişir, hamburger menü simgesine tıklar ve menü bağlantılarının açılır şekilde görünür olduğunu kontrol eder.

**Adımlar:**

1. Mobil görünümde (mobil cihaz simülasyonu ile) tarayıcı başlatılır ve https://automationintesting.online/ adresine gidilir.
2. Hamburger menü simgesi (class="navbar-toggler") bulunur ve tıklanır.
3. Menü elemanlarının bulunduğu alan (class="navbar-collapse") görünür hale gelir.
4. Menü elemanlarının açılır ve kullanıcı tarafından görülebilir olduğu doğrulanır.

**Beklenen Sonuç:**

Hamburger menü simgesine tıklanınca menü açılır ve menü bağlantıları mobil ekranda sorunsuz şekilde görünür olur.

**Test Sonucu:**

Hamburger menü simgesi tıklandığında menü başarıyla açıldı ve görünür oldu. Test başarıyla tamamlandı.

**2.11 Test 11: 404 Sayfası Görünürlük Testi**

**Amaç:**  
Geçersiz veya hatalı bir URL girildiğinde, sitenin kullanıcıya uygun şekilde 404 hata sayfası gösterip göstermediğini doğrulamak.

**Senaryo:**  
Kullanıcı var olmayan bir sayfa adresine erişmeye çalışır ve sistemin 404 hata sayfasını düzgün şekilde görüntülemesi beklenir.

**Adımlar:**

1. Tarayıcı açılır ve https://automationintesting.online/nonexistentpage gibi geçersiz bir URL’ye gidilir.
2. Sayfada bir h1 etiketi bulunur ve içeriğinde "404" veya "not found" ifadelerinden birinin olup olmadığı kontrol edilir.

**Beklenen Sonuç:**

Hatalı URL girildiğinde kullanıcıya 404 hata sayfası gösterilir ve sayfa başlığında "404" veya "not found" ibaresi yer alır.

**Test Sonucu:**

404 hata sayfası başarılı şekilde görüntülendi. Test başarıyla tamamlandı.

**2.12 Test 12: Oda Mevcutluğu ve Yönlendirme Testi**

**Amaç:**

Oda rezervasyonu yapmak için tarihleri seçtikten sonra mevcut odaların listelendiğini ve "Book Now" butonuna basıldığında doğru yönlendirmenin yapıldığını doğrulamak.

**Senaryo:**

Kullanıcı, belirli bir tarih aralığı için oda mevcutluğunu kontrol eder ve ilk oda için "Book Now" butonuna tıklar. Bu durumda, sistem kullanıcının rezervasyon sayfasına yönlendirmesini gerçekleştirmelidir.

**Adımlar:**

1. Chrome tarayıcı başlatılır.
2. https://automationintesting.online/#booking adresine gidilir.
3. Check-in ve Check-out tarihleri belirtilir.
4. "Check Availability" butonuna tıklanır.
5. Mevcut odalar listelendiğinde, ilk oda için "Book Now" butonuna tıklanır.
6. Kullanıcı doğru rezervasyon sayfasına yönlendirilir.

**Beklenen Sonuç:**

Oda mevcutluğu kontrolü sonrasında "Book Now" butonuna tıklayınca rezervasyon sayfasına başarılı bir şekilde yönlendirilmelidir.

**Test Sonucu:**

Yönlendirme işlemi başarılı olduysa test geçer. Yönlendirme gerçekleşmezse test başarısız kabul edilir.