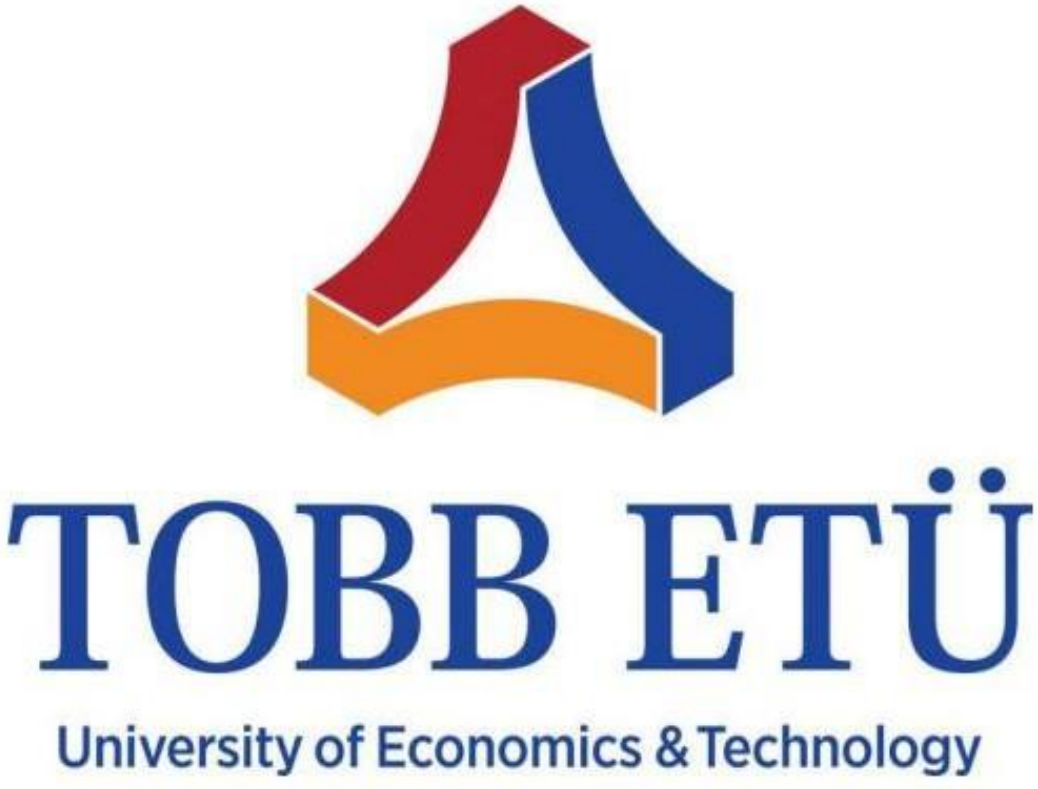
BIL 481 – Acceptance Tests and Criteria



Group Members

Helen Parlar – Artificial Intelligence Engineering

Mete Kılıç – Artificial Intelligence Engineering

Ahmet Çakmak – Medicine & Artificial Intelligence Engineering

Table of Contents

1. Introduction

This document defines Acceptance Criteria and Acceptance Tests for the four core use cases implemented in the Smart Clinical Appointment System. Each test ensures that functional and non-functional requirements are met before the system is demonstrated.

1.1 Test Methodology

- Test Type: Manual Acceptance Testing
- Test Environment: Local Development (localhost)
- Test Data: Pre-seeded database with clinics, doctors, and schedules (via load_slots.py, load_symptoms.py)
- Success Criteria: All acceptance criteria must pass for test to be marked as PASS.

2. Acceptance Test 1: Patient Registration & Login

2.1 Use Case Description

UC-01: A new patient registers for an account and logs in to access the dashboard.

2.2 Acceptance Criteria

Criteria ID	Acceptance Criterion	Success Indicator
AC-1.1	User can successfully register with valid data (full name, email, password)	User record is created in database with role='patient'.

Criteria ID	Acceptance Criterion	Success Indicator
AC-1.2	System validates email format and uniqueness	Duplicate email shows error message.
AC-1.3	Password is securely hashed before storage	Password hash is stored in database (not plaintext).
AC-1.4	JWT access token is generated upon successful login	Token is stored in localStorage.
AC-1.5	User is redirected to correct dashboard based on role	Patient role → /patient/dashboard.
AC-1.6	Success message is displayed after registration	*"Kayıt başarılı! Giriş sayfasına yönlendiriliyorsunuz..."* appears.
AC-1.7	Error messages are displayed for invalid credentials	*"Email veya şifre yanlış"* appears on failed login.

2.3 Traceability

Requirement ID	Design Reference	Implementation
FR1	Section 2.2.2 (Custom User Model)	accounts/models.py
FR2	Section 2.2.2 (JWT Authentication)	accounts/views.py - LoginView
FR1, FR2	Section 2.4.1, 2.4.2 (UI Design)	RegisterPage.jsx, LoginPage.jsx

3. Acceptance Test 2: Direct Polyclinic Selection

3.1 Use Case Description

UC-02: A patient selects a polyclinic directly and verifies their symptoms manually before proceeding to doctor selection.

3.2 Acceptance Criteria

Criteria ID	Acceptance Criterion	Success Indicator
AC-2.1	Polyclinic list loads correctly from database	All 5 polyclinics displayed (Kardiyoloji, KBB, etc.).
AC-2.2	Clicking a polyclinic navigates to symptom verification page	URL changes to /polyclinics/{id}/check/.
AC-2.3	Symptom list is fetched and displayed for selected clinic	At least 3-10 symptoms shown based on clinic.
AC-2.4	"Evet" button navigates to doctor list	URL changes to /select-doctor/{polyId}.
AC-2.5	"Hayır" button navigates to AI symptom input	URL changes to /ai/analyze

3.3 Traceability

Requirement ID	Design Reference	Implementation
FR3, FR4, FR5	Section 3.3 Use Case 2	PolyclinicListPage.jsx
FR4, FR5, FR6	Section 2.4.3 (Symptom Checker UI)	CheckSymptomsPage.jsx
Database Design	Section 2.2.1 (Clinics & Symptoms)	clinics/models.py

4. Acceptance Test 3: AI-Driven Symptom Analysis

4.1 Use Case Description

UC-03: A patient describes their symptoms in free text, and the AI system recommends the most appropriate polyclinic(s).

4.2 Acceptance Criteria

Criteria ID	Acceptance Criterion	Success Indicator
AC-3.1	Free-text input field accepts symptom description	Textarea allows at least 500 characters.
AC-3.2	AI analysis completes within 5 seconds	Response time < 5s (measured via Network tab).
AC-3.3	AI returns valid JSON with clinic recommendations	Response format: {"clinics": ["KBB"], "reason": "..."}.
AC-3.4	System parses JSON and displays recommended clinics	Clinic names shown in UI.
AC-3.5	User can select a recommended clinic	Clicking clinic navigates to doctor list.
AC-3.6	Error handling for API failures	*"AI analizi başarısız"* message shown on timeout.
AC-3.7	Loading indicator shown during AI processing	Spinner or *"Analiz ediliyor..."* text appears.

4.3 Traceability

Requirement ID	Design Reference	Implementation
FR6, FR7	Section 2.2.3 (AI Prompt Logic)	ai/views.py - AIAnalyzeSymptomsView
PR1	Section 3.2 (Performance Requirements)	OpenAI API call optimization
UC-03	Section 3.3 Use Case 3	SymptomInputPage.jsx

5. Acceptance Test 4: Appointment Booking

5.1 Use Case Description

UC-04: A patient selects a doctor, chooses an available time slot, and successfully books an appointment.

5.2 Acceptance Criteria

Criteria ID	Acceptance Criterion	Success Indicator
AC-4.1	Doctor list is filtered by selected polyclinic	Only doctors from chosen clinic appear.
AC-4.2	Doctor information is displayed correctly	Name, title, and clinic shown.
AC-4.3	Clicking "Saat Seç" navigates to slot selection	URL changes to /select-slot/{doctorId}.
AC-4.4	Available slots are loaded and displayed	Date + time shown for each slot.
AC-4.6	Slot locks immediately upon booking	Database is_booked flag updates to TRUE.

Criteria ID	Acceptance Criterion	Success Indicator
AC-4.7	Appointment record is created	New row in appointments_appointment table.
AC-4.8	OneToOne constraint prevents double booking	Attempting to book same slot twice shows "Bu slot dolu" error.
AC-4.9	Success message appears after booking	*"Randevunuz başarıyla oluşturuldu!"*
AC-4.10	User redirected to dashboard	Auto-redirect after 2 seconds.

5.3 Traceability

Requirement ID	Design Reference	Implementation
FR8, FR9	Section 2.2.4 (Slot Locking Logic)	appointments/views.py
FR8	Section 2.4.5 (Doctor Selection UI)	DoctorListPage.jsx, SlotSelectionPage.jsx
TC-01	Quality Assurance Doc	appointments/serializers.py - validate()

6. Acceptance Test 5: Admin & Doctor Management

6.1 Use Case Description

UC-05: An administrator logs in, adds a new doctor to the system, and configures their working hours.

6.2 Acceptance Criteria

Criteria ID	Acceptance Criterion	Success Indicator
AC-5.1	Admin dashboard is accessible only to admins	Non-admins redirected to login or error page.
AC-5.2	Admin can add a new doctor	"Doktor başarıyla oluşturuldu" message appears.
AC-5.3	System enforces @doc.com email constraint	Invalid email format returns error.
AC-5.4	Admin can delete a doctor	"Doktor silindi" confirmation popup appears.
AC-5.5	Admin can add working hours (slots)	New slots appear in the schedule list.

6.3 Traceability

Requirement ID	Design Reference	Implementation
FR13 (Manage Doctors)	Section 2.4.6 (Admin Dashboard)	doctors/views.py - DoctorCreateView
FR14 (Manage Schedule)	Section 2.2.4 (Slot Logic)	doctors/views.py - ScheduleCreateView
SWC-04 (Security Constraint)	Section 2.2.1 (Database Schema)	accounts/models.py - CHECK Constraint

7. Task Matrix

Task	Responsible Member(s)
Define Acceptance Criteria (AC-1.x)	Helen Parlar
Define Acceptance Criteria (AC-2.x, AC-3.x)	Mete Kılıç
Define Acceptance Criteria (AC-4.x)	Ahmet Çakmak
Write Test Steps (AT-01)	Helen Parlar
Write Test Steps (AT-02, AT-03)	Mete Kılıç
Write Test Steps (AT-04)	Ahmet Çakmak
Traceability Mapping	All Members
Screenshot Collection	Ahmet Çakmak, Mete Kılıç
Final Review & Formatting	Helen Parlar