

**Università degli Studi di Salerno**  
Corso di Ingegneria del Software

**FakeBuster  
Test Case Specification  
Versione 2.0**



Data: 12/01/2026

Progetto: FakeBuster	Versione: 2.0
Test Case Specification	Data: 12/01/2026

### Coordinatore del progetto:

Nome	Matricola
Bruno Santo	0512116161
Emiliano Di Giuseppe	0512119155

### Partecipanti:

Nome	Matricola
Bruno Santo	0512116161
Emiliano Di Giuseppe	0512119155

Scritto da:	Bruno Santo & Emiliano Di Giuseppe
-------------	------------------------------------

### Revision History

Data	Versione	Descrizione	Autore
14/12/2025	1.0	Creazione Test Case Specification	Bruno Santo & Emiliano Di Giuseppe
12/01/2026	2.0	Revisione Finale	Bruno Santo & Emiliano Di Giuseppe

# **Indice**

1.	Test case specification identifier.....	4
2.	Test items .....	4
3.	Input specifications .....	5
3.1.	TC_UTE_1: Registrazione Utente .....	5
3.2.	TC_PUB_1: Inserimento Nuovo Post.....	6
3.3.	TC_APP_1: Creazione Appello .....	7
3.4.	TC_SEG_1: Invio Segnalazione .....	7
4.	Output specifications .....	8
4.1.	Output Registrazione (TC_UTE_1) .....	8
4.2.	Output Pubblicazione (TC_PUB_1) .....	9
4.3.	Output Appello (TC_APP_1).....	10
4.4.	Output Segnalazione (TC_SEG_1).....	10
5.	Environmental needs .....	10
6.	Special procedural requirements .....	11
7.	Intercase dependencies .....	11

## 1. Test case specification identifier

**ID Documento:** TCS\_FakeBuster\_v1.0

Questo documento definisce le specifiche dettagliate dei casi di test per il sistema **FakeBuster**

**Social.** Ogni identificativo di test case segue la convenzione definita nel Test Plan:

TC\_[SOTTOSISTEMA]\_[NUM].

Gli identificativi principali coperti in questo documento sono:

- **TC\_UTE\_1:** Registrazione Utente (Gestione Utenza)
- **TC\_UTE\_2:** Login Utente (Gestione Utenza)
- **TC\_PUB\_1:** Inserimento Nuovo Post (Gestione Pubblicazioni)
- **TC\_APP\_1:** Creazione Appello (Gestione Appelli)
- **TC\_SEG\_1:** Creazione Segnalazioni (Gestione Segnalazioni)

## 2. Test items

Gli elementi software (Test Items) oggetto di verifica sono i moduli (Controller) e le entità (Model) definite nell'Object Design Document (ODD).

ID Item	Nome Modulo / Oggetto	Riferimento ODD	Descrizione Funzionale
IT_01	gestione_utenza.py	Sez. 2.2, 3.3	Modulo Flask per la gestione di login e registrazione.
IT_02	gestione_pubblicazioni.py	Sez. 2.2, 3.3	Modulo Flask per la creazione post e interazione con IA.
IT_03	gestione_appelli.py	Sez. 2.2, 3.3	Modulo Flask per la gestione dei ricorsi.
IT_04	Account (Classe)	Sez. 2.1, 3.1	Entità Model che rappresenta l'utente.
IT_05	Post (Classe)	Sez. 2.1, 3.1	Entità Model che rappresenta la notizia e lo score.
IT_06	gestione_segnalazione.py	Sez. 2.2, 3.3	Modulo Flask per la gestione delle segnalazioni.
IT_07	Segnalazione (Classe)	Sez. 2.1, 3.1	Entità Model che rappresenta il report di un utente.

### **3. Input specifications**

In questa sezione vengono dettagliati i valori di input specifici per ogni test case e lo stato iniziale richiesto (Pre-condizioni).

#### **3.1. TC\_UTE\_1: Registrazione Utente**

**Pre-condizioni:** L'utente è un Guest (non autenticato). Database accessibile.

ID Caso	Username	Email	Password	Note
TC_UTE_1.1	(vuoto)	mario@test.it	Pass123!	Input invalido (Campo obbligatorio)
TC_UTE_1.2	Mario	email_errata	Pass123!	Input invalido (Formato Email)
TC_UTE_1.3	Mario	esistente@test.it	Pass123!	Input invalido (Email già nel DB)
TC_UTE_1.4	NuovoUser	nuovo@test.it	PassSicura1!	<b>Input Valido</b>

#### **TC\_UTE\_2: Login Utente**

**Pre-condizioni:** L'utente è un Guest (non autenticato). Database accessibile. Esistono account registrati nel sistema

<b>ID Caso</b>	<b>Username</b>	<b>Password</b>	<b>Ruolo Account</b>	<b>Note</b>
<b>TC_UTE_2.1</b>	(vuoto)	Pass123!	-	<b>Input invalido</b>  (Username mancante - Errore 400/401)
<b>TC_UTE_2.2</b>	NonEsiste	Pass123!	-	<b>Input invalido</b>  (Username non trovato - Errore 401)
<b>TC_UTE_2.3</b>	MarioUser	PassErrata	User	<b>Input invalido</b>  (Password errata - Errore 401)
<b>TC_UTE_2.4</b>	MarioUser	PassGiusta!	User	<b>Input Valido</b>  (Login OK -> Redirect /feed)
<b>TC_UTE_2.5</b>	AdminFC	PassGiusta!	Fact Checker	<b>Input Valido</b>  (Login OK -> Redirect /dashboard)

### 3.2. TC\_PUB\_1: Inserimento Nuovo Post

**Pre-condizioni:** L'utente è autenticato con ruolo 'User'.

ID Caso	Ruolo	Testo Post	Mock AI Score	Note
TC_PUB_1.1	User	(vuoto)	N/A	Input invalido (Testo mancante)
TC_PUB_1.2	User	"La terra è piatta"	0.20	Score < 0.7 (Blocco atteso)
TC_PUB_1.3	User	"Il sole sorge a Est"	0.95	Score >= 0.7 (Pubblicazione attesa)

### 3.3. TC\_APP\_1: Creazione Appello

**Pre-condizioni:** L'utente è autenticato. Esiste un post target nel DB.

ID Caso	ID Post	Stato Post Iniziale	Note
TC_APP_1.1	102	'Bloccato'	Azione Validă

### 3.4. TC\_SEG\_1: Invio Segnalazione

**Pre-condizioni:** L'utente è autenticato come User.

ID Caso	ID Post	Stato Post	Segnalazioni Precedenti (di questo user)	Note
TC_SEG_1.1	103	'Bloccato'	0	Non si può segnalare un post non pubblico
TC_SEG_1.2	104	'Pubblicato'	1	Utente ha già segnalato il post
TC_SEG_1.3	105	'Pubblicato'	0	Azione Validă

## 4. Output specifications

Questa sezione definisce i risultati attesi (Oracolo) che determinano il successo o il fallimento del test.

### 4.1. Output Registrazione (TC\_UTE\_1)

ID Caso	Messaggio Atteso (UI)	Stato Database (Post-Condizione)
TC_UTE_1.1	"Errore: Username obbligatorio"	Nessun record creato in Account.
TC_UTE_1.2	"Errore: Formato email non valido"	Nessun record creato in Account.
TC_UTE_1.3	"Errore: Email già registrata"	Nessun record creato in Account.
TC_UTE_1.5	(Redirect al Feed)	Record creato in Account. Password salvata hashata.

### Output Login (TC\_UTE\_2)

ID Caso	Messaggio Atteso (UI)	Stato Database (Post-Condizione)
TC_UTE_2.1	(Browser) "Compila questo	Nessuna modifica. Sessione non

ID Caso	Messaggio Atteso (UI)	Stato Database (Post-Condizione)
	"campo" / "Credenziali errate"	avviata.
<b>TC_UTE_2.2</b>	"Credenziali errate" (Alert Javascript)	Nessuna modifica. Sessione non avviata.
<b>TC_UTE_2.3</b>	"Credenziali errate" (Alert Javascript)	Nessuna modifica. Sessione non avviata.
<b>TC_UTE_2.4</b>	(Redirect al Feed)	Nessuna modifica al DB. Sessione utente attiva ( <code>user_id</code> popolato).
<b>TC_UTE_2.5</b>	(Redirect alla Dashboard)	Nessuna modifica al DB. Sessione Fact Checker attiva.

#### 4.2. Output Pubblicazione (TC\_PUB\_1)

ID Caso	Messaggio Atteso	Stato Database (Post.stato)
<b>TC_PUB_1.2</b>	"Errore: Testo obbligatorio"	Nessun record creato.
<b>TC_PUB_1.3</b>	"Contenuto bloccato per bassa attendibilità"	Record creato con stato ' <b>Bloccato</b> '.
<b>TC_PUB_1.4</b>	"Post pubblicato con successo"	Record creato con stato ' <b>Pubblicato</b> '.

#### 4.3. Output Appello (TC\_APP\_1)

ID Caso	Messaggio Atteso	Stato Database (Appello / Post)
TC_APP_1.1	"Appello inviato con successo"	Record in Appello. Appello aggiornato a 'aperto'.

#### 4.4. Output Segnalazione (TC\_SEG\_1)

ID Caso	Messaggio Atteso	Stato Database (Segnalazione)
TC_SEG_1.1	"Errore: Il post non è disponibile"	Nessun record creato.
TC_SEG_1.2	"Errore: Hai già segnalato questo post"	Nessun nuovo record creato.
TC_SEG_1.3	"Segnalazione inviata con successo"	Nuovo record creato in Segnalazione.

### 5. Environmental needs

Specifiche dell'ambiente hardware e software necessarie per l'esecuzione dei test, in coerenza con l'architettura definita nel **System Design Document (SDD)**.

- **Hardware:**
  - **Server:** Minimo 4 vCPU, 8GB RAM (ambiente di staging/test).
  - **Client:** Workstation standard con accesso di rete al server.
- **Software:**
  - **OS:** Linux (Ubuntu 20.04) o Windows 10/11.

- **Backend Framework:** Python 3.9+ con **Flask**.
- **Database:** MySQL Community Server 8.0.
- **Test Runner:** Framework **PyTest** o **Unittest** per l'esecuzione automatizzata.
- **Browser:** Google Chrome (ultima versione) per test E2E con Selenium.

## 6. Special procedural requirements

Procedure speciali richieste per la preparazione dell'ambiente di test:

1. **AI Mocking:** Per i test funzionali (TC\_PUB\_1.3, TC\_PUB\_1.4), il servizio IA reale non deve essere invocato. Utilizzare librerie di mocking (es. unittest.mock) per simulare la risposta dello score (0.20 o 0.95) ed evitare latenze di rete o costi computazionali.
2. **Database Fixtures:** Prima dell'esecuzione della suite di test, il database deve essere resettato a uno stato noto (pulizia tabelle Account, Post, Appello) per garantire la ripetibilità dei test (es. evitare errori di chiave primaria duplicata).
3. **Configurazione Ruoli:** Assicurarsi che nel database di test siano pre-popolati almeno un utente FactChecker e un utente User standard.

## 7. Intercase dependencies

Dipendenze logiche tra i casi di test:

- **TC\_PUB\_1 dipende da TC\_UTE\_1:** Il test di pubblicazione (TC\_PUB\_1) richiede un utente autenticato. Pertanto, il test di registrazione (TC\_UTE\_1.5) deve essere completato con successo (o deve esistere un utente pre-caricato) prima di procedere.
- **TC\_APP\_1 dipende da TC\_PUB\_1:** Il test di creazione appello (TC\_APP\_1.3) richiede l'esistenza di un post in stato 'Bloccato'. Pertanto, il test TC\_PUB\_1.3 (Inserimento post con score basso) è un prerequisito per l'esecuzione dei test sugli appelli.
- **TC\_SEG\_1 dipende da TC\_PUB\_1:** Il test di invio segnalazione (TC\_SEG\_1.3) richiede l'esistenza di un post in stato 'Pubblicato' di un altro utente. Pertanto, il test TC\_PUB\_1.4 (Pubblicazione con successo) è un prerequisito per l'esecuzione dei test sulle segnalazioni.