

**FakeBuster**  
**Requirements Analysis Document**  
**Versione 3.0**



Data: 09/11/2025

Progetto: FakeBuster	Versione: 3.0
Documento: Requirements Analysis Document	Data: 09/11/2025

### Coordinatore del progetto:

Nome	Matricola
Bruno Santo	0512116161
Emiliano Di Giuseppe	0512119155

### Partecipanti:

Nome	Matricola
Bruno Santo	0512116161
Emiliano Di Giuseppe	0512119155

<b>Scritto da:</b>	Bruno Santo & Emiliano Di Giuseppe
--------------------	------------------------------------

## Revision History

Data	Versione	Descrizione	Autore
25/10/2025	1.0	Creazione e modifica rad prima parte	Bruno Santo & Emiliano Di Giuseppe
27/10/2025	2.0	Modifica casi d'uso , scenari , diagrammi	Bruno Santo & Emiliano Di Giuseppe
09/11/2025	3.0	Object model	Bruno Santo & Emiliano Di Giuseppe

# Indice

1.	Introduction .....	4
1.1.	Purpose of the system.....	4
1.2.	Scope of the system.....	4
1.3.	Objectives and success criteria of the project .....	4
1.4.	Definitions, acronyms, and abbreviations.....	4
1.5.	References .....	4
2.	Current system.....	5
3.	Proposed system .....	5
3.1.	Overview .....	5
3.2.	Functional requirements.....	5
3.3.	Nonfunctional requirements.....	6
3.3.1.	Usability .....	6
3.3.2.	Reliability.....	6
3.3.3.	Performance .....	6
3.3.4.	Supportability.....	6
3.3.5.	Implementation .....	6
3.3.6.	Interface .....	6
3.3.7.	Packaging.....	7
3.3.8.	Legal.....	7
3.4.	System models .....	7
3.4.1.	Scenarios .....	7
3.4.2.	Use case model.....	11
3.4.3.	Object model.....	19
3.4.4.	Dynamic Model .....	22
3.4.5.	User interface-navigational paths and screen mock-ups .....	26

## **1. Introduction**

### **1.1.Purpose of the system**

FakeBuster Social nasce come piattaforma sperimentale che unisce intelligenza artificiale e partecipazione umana per creare un ecosistema informativo più affidabile.

L'obiettivo principale è ridurre la diffusione di notizie false alla radice, bloccandone la pubblicazione già in fase di inserimento.

### **1.2.Scope of the system**

Il progetto riguarda lo sviluppo di un social network controllato, in cui ogni contenuto testuale viene valutato tramite modelli di AI di verifica automatica.

Il sistema consente:

- la pubblicazione di notizie solo dopo la validazione;
- la tracciabilità di ogni decisione automatica tramite dei verificatori umani;
- l'analisi delle decisioni dell'IA per migliorare il modello.

### **1.3.Objectives and success criteria of the project**

Il progetto FakeBuster Social mira a:

- Creare un social network dove la qualità dell'informazione è prioritaria.
- Sviluppare un modello di AI capace di stimare in tempo reale la veridicità dei testi.

Criteri di successo:

- Almeno l'75% di accuratezza nella classificazione di notizie veritiere o false.
- Tempo di risposta dell'IA inferiore a 5 secondi.
- Interfaccia accessibile e facile da usare.
- Garantire revisione umana

### **1.4. Definitions, acronyms, and abbreviations**

- IA / AI (Intelligenza Artificiale) – Tecnologia che consente al sistema di analizzare e valutare automaticamente le notizie.
- Fake news – Notizia falsa o manipolata, diffusa con intento ingannevole.
- Fact-checker – Utente esperto incaricato di verificare la correttezza delle decisioni automatiche del sistema in caso di appello o segnalazione del post
- User - Utente autenticato che può sia pubblicare che leggere post

### **1.5.References**

- Object-Oriented Software Engineering Using UML, Patterns, and Java™ Third Edition - Bernd Bruegge & Allen H. Dutoit
- Slide del corso di ingegneria del software di Andre De Lucia

## 2. Current system

Attualmente, la diffusione di notizie sui social network avviene senza un controllo sistematico della veridicità dei contenuti.

Le piattaforme più diffuse (come X, Facebook o Instagram) si basano su meccanismi di segnalazione manuale o su algoritmi di moderazione generici, che intervengono solo dopo la pubblicazione del contenuto.

Questo approccio presenta diverse problematiche:

- Le fake news possono diffondersi rapidamente prima che vengano rimosse.
- Gli algoritmi di moderazione spesso non distinguono il contesto o l'intento del messaggio.
- La fiducia degli utenti nelle fonti digitali è in costante calo.
- Non esiste una tracciabilità trasparente delle decisioni automatizzate.

Attualmente, quindi, il controllo dell'informazione è reattivo e non preventivo, lasciando spazio a manipolazioni e disinformazione.

## 3. Proposed system

### 3.1. Overview

FakeBuster Social propone un nuovo paradigma di social network, in cui la pubblicazione di una notizia è subordinata a una valutazione automatica di attendibilità effettuata da un modello di intelligenza artificiale.

Il sistema integra inoltre una supervisione umana (fact-checker) che analizza eventuali ricorsi o segnalazioni degli user

Funzionalità principali:

- Analisi automatica del testo prima della pubblicazione.
- Classificazione dei contenuti in *attendibili*, *sospetti* o *falsi*.
- Pannello dedicato ai fact-checker per la revisione e l'addestramento continuo del modello.

### 3.2. Functional requirements

#### Funzionalità per l'Utente (Publisher / Reader)

- L'utente può registrarsi e accedere tramite l'apposito form
- Ricevere conferma immediata di ricezione.
- Presentare appello su post bloccati.
- Segnalare altri post (report) con motivo e commento.
- Ricevere notizie sullo stato del post (pubblicato, bloccato, in revisione, esito appello).
- Creare nuovo post: titolo, testo, link e allegati.

#### Funzionalità dell'AI Service / Processo automatico

- Valutazione automatica per ogni post

- SLA: supporto alla pubblicazione automatica entro 5s quando previsto.

#### **Funzionalità per il FactChecker**

- Accedere alla dashboard di revisione con visuale unificata: testo, score IA e segnali utente. Può anche vedere le segnalazioni e i post bloccati
- Azioni possibili su un caso: pubblica, mantieni bloccato, rimuovi, edit.

#### **Operazioni di pubblicazione e stato**

- Stati supportati: in attesa, pubblicato, bloccato, rimosso.
- Transizioni determinate da regole: IA + soglie + azione FactChecker.
- Azioni automatiche visibili al pubblico (banner + motivo sintetico) quando applicate.

#### **Requisiti di qualità chiave (breve)**

- Decisione e pubblicazione automatica entro 5s quando applicabile.
- Notifica all'utente (es. blocco) entro 5s.

### **3.3. Nonfunctional requirements**

#### **3.3.1. Usability**

Interfacce chiare per invio post, appello e coda fact-checker; dashboard unificata per fact-checker; conformità base a standard di accessibilità.

#### **3.3.2. Reliability**

Il sistema deve garantire un'accuratezza minima dell'75% nelle classificazioni AI.

#### **3.3.3. Performance**

Decisione IA + pubblicazione automatica  $\leq 10s$  quando prevista

#### **3.3.4. Supportability**

Tutte le modifiche devono essere tracciate e versionate.

#### **3.3.5. Implementation**

- Linguaggio principale: Python (per AI e backend).
- Framework web: Flask / FastAPI.
- Database: MySQL.
- Frontend: html/css

#### **3.3.6. Interface**

- Utente: interfaccia web responsive.

- Fact-checker: pannello dedicato per gestione dei ricorsi e segnalazioni
- API REST: per comunicazione tra moduli AI e piattaforma social.

### 3.3.7. Packaging

Il sistema sarà distribuito come piattaforma web modulare, con componenti separati per backend, motore AI, frontend .

Dal lato utente, il sistema sarà accessibile tramite interfaccia web.

### 3.3.8. Legal

Il trattamento dei dati personali seguirà il Regolamento GDPR (UE 2016/679).

Tutti i contenuti generati dagli utenti saranno soggetti a una policy di responsabilità che vieta la pubblicazione di materiale offensivo, discriminatorio o lesivo di diritti altrui.

## 3.4. System models

### 3.4.1. Scenarios

#### Scenario 1 — login

##### Attori Partecipanti

maria:guest

##### Flusso di eventi

1. Maria apre la home page
2. Maria clicca “accedi”.
3. Inserisce credenziali
4. Il sistema la riconosce come user e non fact checker e la reindirizza al feed
5. Maria vede post degli altri e può pubblicarne dei suoi

##### Precondizione

utente non è autenticato e possiede un account valido.

##### Postcondizione

utente è autenticato e ha accesso alle funzionalità del suo ruolo

##### Quality requirements

- Messaggio di errore generico (es. "Credenziali non valide") in caso di errore

## Scenario 2 — registrazione

### Attori Partecipanti

maria guest

### Flusso di eventi

1. Maria apre la home page
2. Maria clicca “registrazione”.
3. Inserisce username, mail e password,
4. Il sistema controlla se l’email o l’username già esiste, se tutti i controlli sono superati crea l’utente
5. Maria viene reindirizzata al feed

### Precondizione

L’utente non ha un account

L’indirizzo email non è già associato a un altro account.

### Postcondizione

L’account è creato e attivo.

### Quality requirements

- Password memorizzata solo in forma hashata

## Scenario 3 — userSubmitsNews

### Attori Partecipanti

maria:User, AIService

### Flusso di eventi

6. Maria apre la form “Nuovo Post” e incolla titolo + testo + eventuale immagine.
7. Maria clicca “Invia”.
8. AIService analizza il testo: estrae features, calcola embedding e produce una stima di attendibilità (score = 0.87).
9. AIService invia la decisione e il log (features principali, score, timestamp) a FakeBusterSystem.
10. FakeBusterSystem, con soglia predefinita (es.  $\text{publish\_if\_score} \geq 0.7$ ), pubblica automaticamente il post nello stream pubblico.
11. Maria riceve notifica “Post pubblicato” con link.

### Precondizione

L’utente è autenticato e non supera vincoli di rate-limit.

### Postcondizione

Il post è pubblicato e la decisione dell’IA è registrata.

### Quality requirements

- Decisione e pubblicazione entro 5 secondi.

## **Scenario 4 — aiBlocksSubmission**

### **Attori partecipanti**

luca:User,AIService

### **Flusso di eventi**

1. Luca invia una notizia (titolo + testo + immagine).
2. AIService valuta e produce score = 0.18 (basso).
3. FakeBusterSystem cambia stato del post in bloccato e invia notifica a Luca: “Il tuo contenuto non è pubblicato per possibile bassa attendibilità. Puoi modificare e riprovare o presentare appello.”
4. Luca può modificare il testo o allegare fonti; se modifica e reinvia, il ciclo riparte dalla valutazione AI.

### **Precondizione**

Utente autenticato; contenuto analizzabile (testo o testo+link).

### **Postcondizione**

Post rimane bloccato o viene reinviato dopo modifica.

### **Quality requirements**

- Notifica all’utente entro 5s.
- Motivo sintetico (es. “score 0.18 — alta probabilità di contenuto non verificato”) mostrato all’utente; non esporre feature sensibili.

## **Scenario 5 — userAppeal**

### **Attori partecipanti**

anna:User, fact checker:moderator1,AIService

### **Flusso di eventi**

1. Anna trova il suo contenuto bloccato e clicca “Presenta appello”
2. FakeBusterSystem registra l’appello e mette in coda per revisione umana.
3. moderator1 apre l’interfaccia che mostra: testo del post, score IA.
4. Moderator1 valuta e sceglie fra: pubblica, mantieni bloccato.
5. Se pubblica → FakeBusterSystem pubblica il post. Se mantieni bloccato → notifica all’utente con motivazione sintetica.

### **Precondizione**

Post in stato revisione e appello inviato.

### **Postcondizione**

Post pubblicato o definitivamente respinto.

### **Quality requirements**

- Fact checker deve visualizzare tutte le informazioni rilevanti (testo, log IA) in un’unica schermata.

### **Scenario 6 — userReportsPublicPostToFactChecker**

#### **Attori partecipanti**

user maria , , fact checker:moderator1

#### **Flusso di eventi**

1. Maria clicca **Segnala** su un post
2. system crea report.
3. system assegna la task a moderator1 e invia notifica.
4. , fact checker:moderator1 valuta e decide.
5. Autore può vedere la decisione del fact checker guardando i suoi post.

#### **Precondizione**

utente autenticato + post pubblico visibile.

#### **Postcondizione**

post bloccato o pubblico a discrezione del fact checker

#### **Quality requirements**

trasparenza pubblica per azioni automatiche (banner + motivo);

### **Scenario 7 — userViewsPostHistory**

#### **Attori partecipanti**

user maria

#### **Flusso di eventi**

1. maria clicca sul proprio profilo → seleziona “I miei post”.
2. system recupera l'elenco dei post associati all'utente.
3. system mostra l'elenco [titolo, stato (Pubblicato, Bloccato, In Revisione)].

#### **Precondizione**

utente autenticato.

#### **Postcondizione**

utente visualizza il proprio storico post.

#### **Quality requirements**

Accuratezza dello stato (lo stato visualizzato deve rispecchiare lo stato reale del post).

### **Scenario 8 — factCheckerViewsAppealsQueue**

#### **Attori partecipanti**

factChecker moderator1

#### **Flusso di eventi**

- 1 moderator1 esegue il login .,
- 2 moderator1 naviga alla sezione "Appelli da Revisionare".,
- 3 system recupera l'elenco di tutti i post in stato "Appello Inviato" (generati da Scenario 3).,
- 4 system mostra l'elenco [Titolo Post, autore].,
- 5 moderator1 seleziona un item dall'elenco blocca o pubblica

#### **Precondizione**

factChecker è autenticato e ha i permessi di revisione.

**Postcondizione**

factChecker visualizza la lista di tutti gli appelli pendenti.

**Quality requirements**

- La lista deve essere ordinabile (es. per data, per score).,
- La lista deve aggiornarsi in tempo reale o con un refresh manuale.,

**Scenario 9 — factCheckerViewsReportsQueue**

**Attori partecipanti**

factChecker moderator1

- 1 moderator1 esegue il login (Scenario 7).,
- 2 moderator1 naviga alla sezione "Segnalazioni Pubbliche".,
- 3 system recupera l'elenco di tutti i post segnalati (generati da Scenario 4).,
- 4 system mostra l'elenco [Titolo Post,autore].,
- 5 moderator1 seleziona un item dall'elenco blocca o pubblica

**Precondizione**

factChecker è autenticato e ha i permessi di revisione.

**Postcondizione**

factChecker visualizza la lista di tutte le segnalazioni pendenti.

**Quality requirements**

- La lista deve dare priorità ai post con segnalazioni multiple.

**3.4.2. Use case model**

**UC1 — userSubmitsNews**

**Nome caso d'uso Pubblicazione di una notizia**

**Attori principali** User , AI

**Descrizione** L'utente invia una nuova notizia che viene analizzata automaticamente dall'AI e pubblicata se supera la soglia di attendibilità. (mock-up 6)

**Precondizione** L'utente è autenticato.  
1. L'utente apre la form "Nuovo post" e inserisce titolo, testo e immagine.  
2. Clicca "Invia".  
3. L'AI analizza il contenuto e assegna uno *score* di attendibilità.

**Flusso principale** 4. FakeBusterSystem confronta lo score con la soglia.  
5. Se  $\text{score} \geq 0.7 \rightarrow$  pubblica il post.  
6. L'utente riceve la notifica "Post pubblicato".

**Postcondizione** Il post è pubblicato e la decisione dell'AI è registrata nel log.

**Requisiti di qualità** Tempo massimo di analisi e pubblicazione: 5s.

**Flusso di eventi alternativi:** Flusso Alternativo Questo flusso inizia al punto 5 del Flusso Principale.5a. Se score < 0.7 (es. 0.18):6a. FakeBusterSystem blocca la pubblicazione. FakeBusterSystem invia una notifica pop-up all'utente (mock-up 6)

## UC2 — userAppeal

<b>Nome caso d'uso</b>	<b>Appello di un contenuto bloccato</b>
<b>Attori principali</b>	User , FactChecker, AI
<b>Descrizione</b>	L'utente contesta la decisione automatica di blocco e chiede la revisione manuale da parte di un fact checker. (mock-up 6)
<b>Precondizione</b>	Post in stato "bloccato"; appello inviato. (mock-up 5)
<b>Flusso principale</b>	<ol style="list-style-type: none"> <li>1. L'utente clicca "Presenta appello".</li> <li>2. Il sistema registra l'appello e lo invia in coda.</li> <li>3. Il fact checker apre l'interfaccia di revisione (mock-up 7)</li> <li>4. Valuta il caso e sceglie "Pubblica" o "blocca".</li> <li>5. Il sistema esegue la decisione e invia notifica all'utente.</li> </ol>
<b>Postcondizione</b>	Il post viene pubblicato o respinto definitivamente.
<b>Requisiti di qualità</b>	Tutte le info visibili in un'unica schermata.

### UC3 — userReportsPublicPostToFactChecker

**Nome caso d'uso** Segnalazione di un post pubblico

**Attori principali** User, FactChecker, AIService

**Descrizione** Un utente segnala un post pubblico sospetto. Il fact checker valuta la segnalazione e decide se mantenere o bloccare il post. (mock-up 4)

**Precondizione** L'utente è autenticato; il post è pubblico.

**Flusso principale**

1. L'utente clicca "Segnala" su un post.
2. Il sistema crea un report e lo assegna a un fact checker.
3. Il fact checker analizza il post e decide. (mock-up 7)
4. L'autore visualizza l'esito della revisione.

**Postcondizione** Il post rimane pubblico o viene bloccato.

**Requisiti di qualità** Trasparenza: banner con motivo della revisione visibile al creatore del post.

### UC4 — userReportsPublicPostToFactChecker

**Nome caso d'uso** Visualizza Cronologia Post

**Attori principali** User

**Descrizione** L'utente accede alla propria area personale per visualizzare l'elenco di tutti i post che ha inviato e il loro stato corrente (es. Pubblicato, Bloccato, In revisione) (mock-up 5)

**Precondizione** L'utente è autenticato;

**Flusso principale**

1. L'utente clicca sul suo profilo e seleziona "I miei post".
2. Compila il motivo della segnalazione.
3. Il sistema mostra l'elenco, indicando per ciascuno il titolo e lo stato (Pubblicato, Bloccato, In Revisione).

**Postcondizione** L'utente ha visualizzato il suo storico.

**Requisiti di qualità** Trasparenza: banner con stato post , Accuratezza dello stato (lo stato visualizzato deve rispecchiare lo stato reale del post).

## UC5 — login

<b>Nome caso d'uso</b>	<b>Login</b>
<b>Attori principali</b>	Guest (utente non autenticato)
<b>Descrizione</b>	Un utente guest accede alla piattaforma inserendo le proprie credenziali. Il sistema verifica l'identità e, se l'autenticazione ha successo, lo reindirizza al feed principale in base al suo ruolo (User o Fact Checker).
<b>Precondizione</b>	L'utente non è autenticato. L'utente possiede un account valido e attivo.
<b>Flusso principale</b>	<ol style="list-style-type: none"><li>1. il guest va sulla home e clicca accedi (mock-up 1-3)</li><li>2. Inserisce mail e password</li><li>3. Il sistema invia le credenziali al modulo di autenticazione. Se le credenziali sono valide: Se l'utente è User, viene reindirizzato al feed pubblico dove può visualizzare i post e pubblicarne di nuovi. Se l'utente è Fact Checker, viene reindirizzato alla dashboard di verifica.</li><li>4. se le credenziali sono non valide: Il sistema mostra un messaggio di errore generico ("Credenziali non valide"). L'utente può riprovare o scegliere "Password dimenticata?".</li></ol>
<b>Postcondizione</b>	L'utente è autenticato e ha accesso alle funzionalità del proprio ruolo. L'evento di login è registrato nel log di sistema.
<b>Requisiti di qualità</b>	Messaggio di errore sempre generico , per segnalare errori del guest

## UC6 — registrazione

<b>Nome caso d'uso</b>	<b>registrazione</b>
<b>Attori principali</b>	Guest (utente non autenticato)
<b>Descrizione</b>	Un utente guest crea un nuovo account inserendo username, email e password. Il sistema valida i dati, verifica che non esistano duplicati e crea l'account, permettendo l'accesso immediato al feed.
<b>Precondizione</b>	L'utente non è autenticato. L'indirizzo email e lo username non risultano già associati a un altro account.
<b>Flusso principale</b>	<ol style="list-style-type: none"><li>1. il guest va sulla home e clicca registrati (mock-up 1-2)</li><li>2. Inserisce i dati richiesti: username, email e password.</li><li>3 Il sistema valida i campi:<ul style="list-style-type: none"><li>Verifica il formato corretto dell'email.</li><li>Controlla che la password rispetti i requisiti minimi (lunghezza, complessità).</li><li>Controlla che lo username non contenga caratteri non ammessi.</li></ul></li><li>Il sistema verifica che email e username non siano già presenti nel database.</li><li>4. Se email o username esistono già mostra un messaggio di errore: "Email o username già registrati." e richiede nuovi dati.</li><li>5. Se i controlli sono tutti superati, procede con la creazione dell'account.</li><li>6. Il sistema registra il nuovo utente nel database, memorizzando la password in forma hashata.</li><li>7. L'utente viene reindirizzato al feed principale e può iniziare a utilizzare la piattaforma. (mock-up 5)</li></ol>
<b>Postcondizione</b>	L'account è creato e attivo. L'utente è autenticato e ha accesso alle funzionalità base. L'evento di registrazione è registrato nel sistema.
<b>Requisiti di qualità</b>	Messaggio di errore sempre generico , per segnalare errori del guest Password memorizzata solo in forma hashata

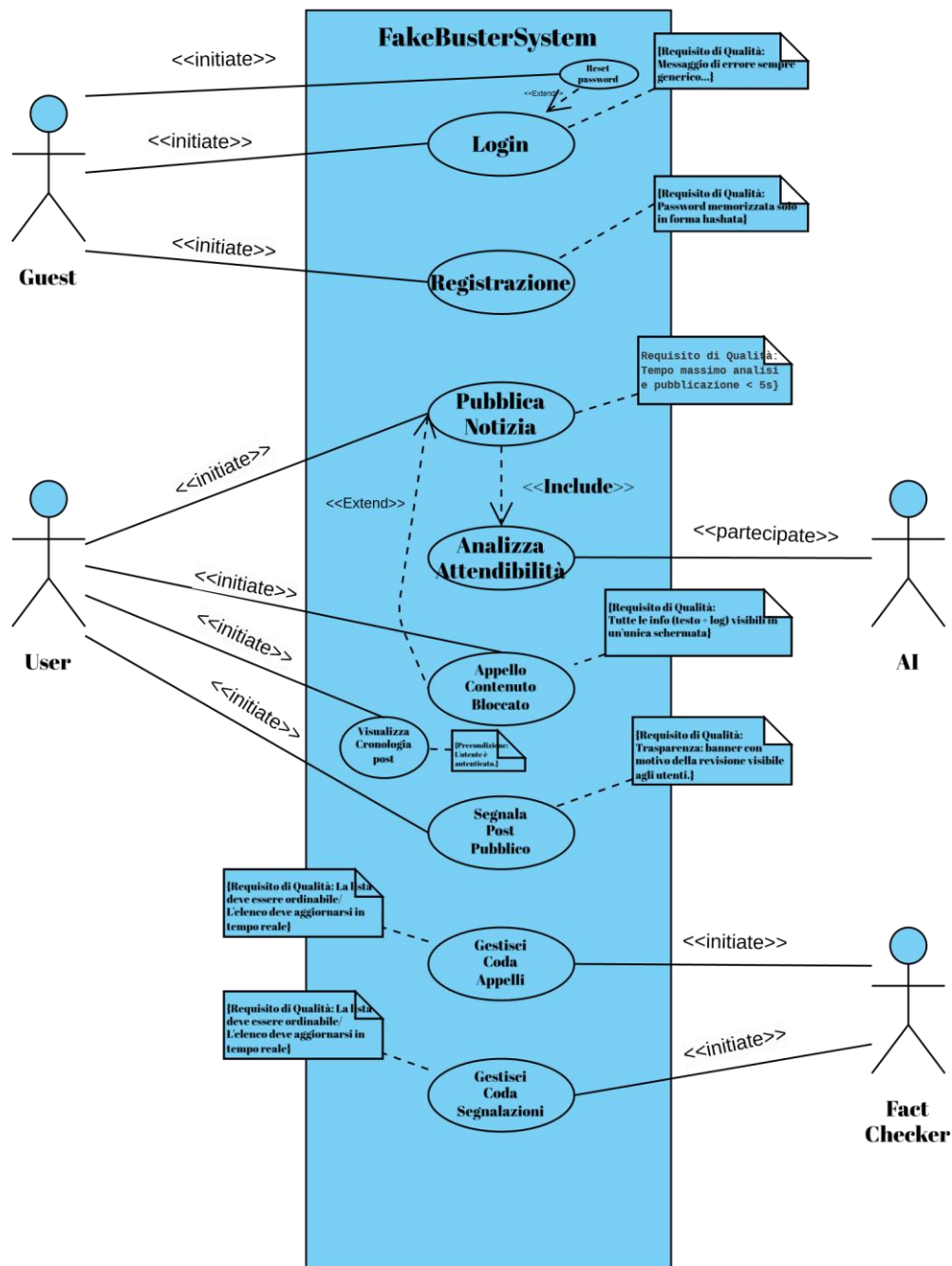
## UC7 — factCheckerViewsAppealsQueue

<b>Nome caso d'uso</b>	<b>Visualizzazione e gestione della coda di appelli</b>
<b>Attori principali</b>	FactChecker
<b>Descrizione</b>	<p>Il FactChecker accede alla sezione dedicata agli appelli per visualizzare tutti i post in stato “Appello Inviato”.</p> <p>Può analizzare ciascun caso e decidere se bloccare o pubblicare il post, chiudendo così la revisione.</p>
<b>Precondizione</b>	<p>Il FactChecker è autenticato.</p> <p>L'utente dispone dei permessi di revisione (ruolo = FactChecker).</p>
<b>Flusso principale</b>	<ol style="list-style-type: none"><li>1. Il FactChecker esegue il login nel sistema.</li><li>2. il sistema lo reindirizza alla dashboard (mock-up 7)</li><li>3 il factChecker vede nella sezione appelli gli appelli con autore , titolo , testo.</li><li>4. Il FactChecker può scegliere due opzioni per ogni appello, pubblica o rimuovi</li><li>5 . Il sistema, se la il FactChecker cliccherà “pubblica”, pubblicherà il post e nella pagina dei post ,l'autore vedrà sul suo post la targhetta pubblicato . In caso contrario se cliccherà “blocca” il sistema non pubblica il post e l'autore vedrà sul suo post la targhetta bloccato (mock-up 5)</li></ol>
<b>Postcondizione</b>	<p>L'account è creato e attivo.</p> <p>Il FactChecker visualizza e può gestire la lista aggiornata di tutti gli appelli pendenti.</p>
<b>Requisiti di qualità</b>	L'elenco deve aggiornarsi in tempo reale o con un refresh manuale.

## UC8 — factCheckerViewsReportQueue

<b>Nome caso d'uso</b>	<b>Visualizzazione e gestione della coda delle segnalazione</b>
<b>Attori principali</b>	FactChecker
<b>Descrizione</b>	Il FactChecker accede alla sezione dedicata alle segnalazioni pubbliche per esaminare tutti i post segnalati dagli utenti. Può visualizzare l'elenco dei post, analizzarne i dettagli e decidere se bloccare o pubblicare ciascun contenuto segnalato.
<b>Precondizione</b>	Il FactChecker è autenticato. L'utente dispone dei permessi di revisione (ruolo = FactChecker).
<b>Flusso principale</b>	<ol style="list-style-type: none"><li>1. Il FactChecker esegue il login nel sistema.</li><li>2. il sistema lo reindirizza alla dashboard (mock-up 7)</li><li>3 il factChecker vede nella sezione segnalazioni , le segnalzioni con autore , titolo , testo.</li><li>4. Il FactChecker può scegliere due opzioni per ogni appello, safe o rimuovi</li><li>5 . Il sistema, se la il FactChecker cliccherà “safe”, il post rimarrà pubblico. In caso contrario se cliccherà “rimuovi” il sistema rimuovi il post e l'autore vedrà sul suo post la targhetta bloccato (mock-up 5)</li></ol>
<b>Postcondizione</b>	L'account è creato e attivo. Il FactChecker visualizza e può gestire la lista aggiornata di tutti gli appelli pendenti.
<b>Requisiti di qualità</b>	La lista degli appelli deve essere ordinabile (per data, score, autore). L'elenco deve aggiornarsi in tempo reale o con un refresh manuale.

## diagramma use case



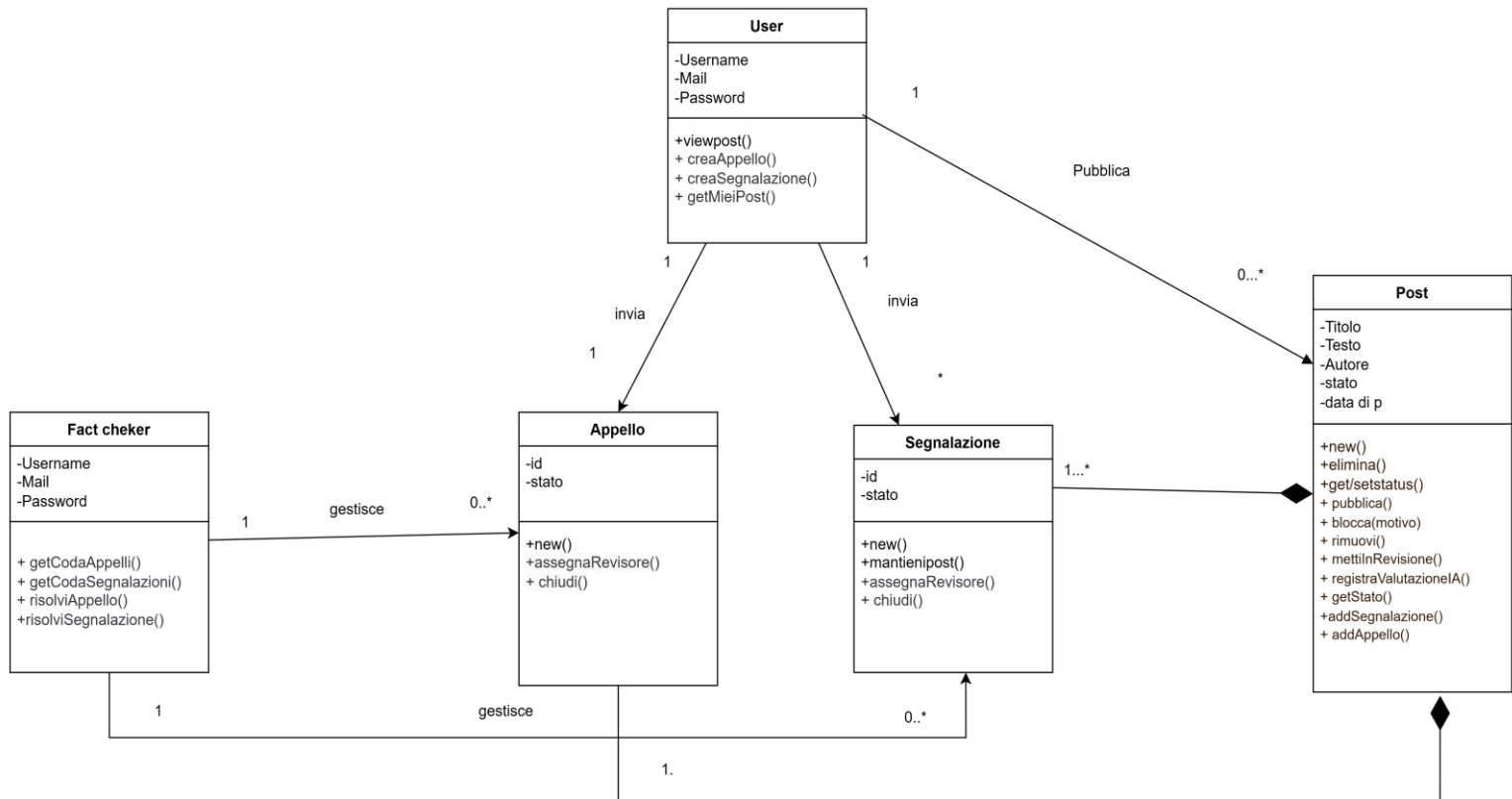
### 3.4.3. Object model

Tabella oggetti

Nome	Tipo	Descrizione
User	Entity	L'utente registrato che può pubblicare notizie , segnalare altri post e presentare appelli per i propri post bloccati
Fact-checker	Entity	L'utente esperto (moderatore) che ha il permesso di accedere a una dashboard per analizzare e prendere decisioni su appelli e segnalazioni.
AI	Entity	Rappresenta il sistema di Intelligenza Artificiale che agisce come un componente per analizzare e valutare l'attendibilità dei post.
Post	Entity	L'entità dati centrale; memorizza il contenuto inviato dall'utente (titolo, testo, ecc. ) e il suo stato corrente (es. 'in attesa', 'pubblicato', 'bloccato' ).
Appello	Entity	L'entità dati che rappresenta la richiesta formale di revisione manuale, creata quando un utente contesta il blocco di un suo post.
Segnalazione	Entity	L'entità dati che rappresenta il "report" inviato da un utente per notificare un post pubblico che ritiene sospetto o falso.
Homepage	Boundary	L'interfaccia (schermata) iniziale per gli utenti non autenticati (Guest), che presenta le opzioni di Login e Registrazione
Registrazione	Boundary	L'interfaccia (form) che raccoglie i dati (username, email, password ) per consentire a un Guest di creare un nuovo account.

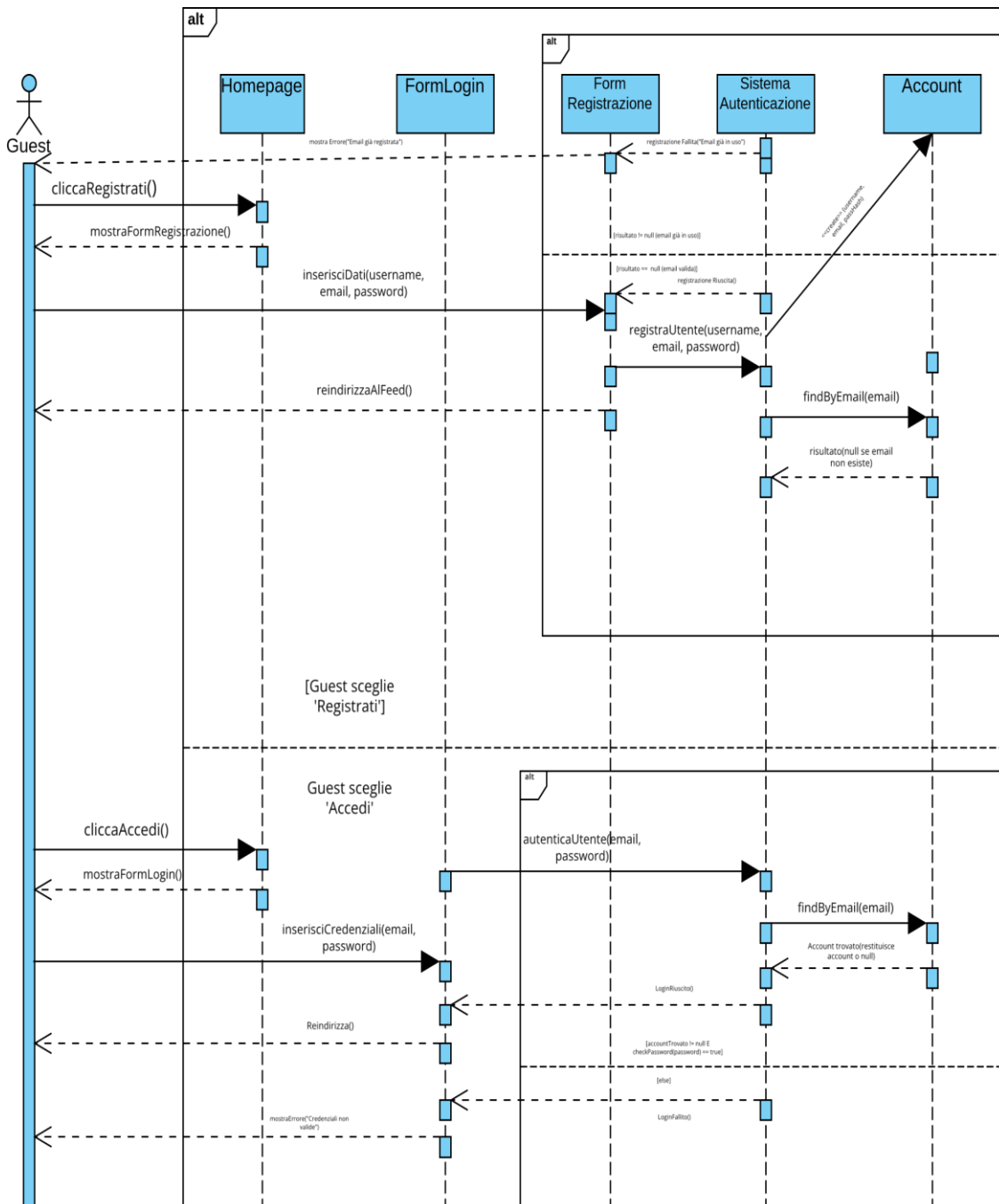
Login	Boundary	L'interfaccia (form) che raccoglie le credenziali (email, password ) per permettere a un Guest di autenticarsi
Feed	Boundary	L'interfaccia (bacheca) principale dove un User visualizza l'elenco dei post pubblicati dagli altri.
PubblicaPost	Boundary	L'interfaccia (form) utilizzata dall' User per inserire titolo, testo e allegati di una nuova notizia da sottoporre ad analisi.
PaginaMieiPost	Boundary	L'interfaccia (schermata) personale dove l' User visualizza lo storico dei propri post e il loro stato corrente (Pubblicato, Bloccato, ecc.)
DashBoard FactChecker	Boundary	L'interfaccia (pannello di controllo) riservata al Fact-checker per visualizzare le code e gestire gli appelli e le segnalazioni pendenti.
SistemaAutenticazione	Control	La classe che gestisce la logica di business per la registrazione (validando i dati ) e il login (verificando le credenziali ).
Gestore Pubblicazioni	Control	La classe che coordina il processo di invio di un post: riceve i dati dal form, invoca l' AiService , interpreta lo score e aggiorna lo stato del Post (pubblicato o bloccato).
GestoreAppelli	Control	La classe che gestisce la logica di un appello: crea l'entità Appello, la inserisce nella coda per il Fact-checker e applica la decisione finale (pubblica o mantieni bloccato).
GestoreSegnalazioni	Control	La classe che gestisce la logica di una segnalazione: crea l'entità Segnalazione, la assegna a un Fact-checker e applica la sua decisione (rimuovi o "safe").
AiService	Control	Il componente software (spesso un'API ) che agisce da "colla", prendendo il testo del post

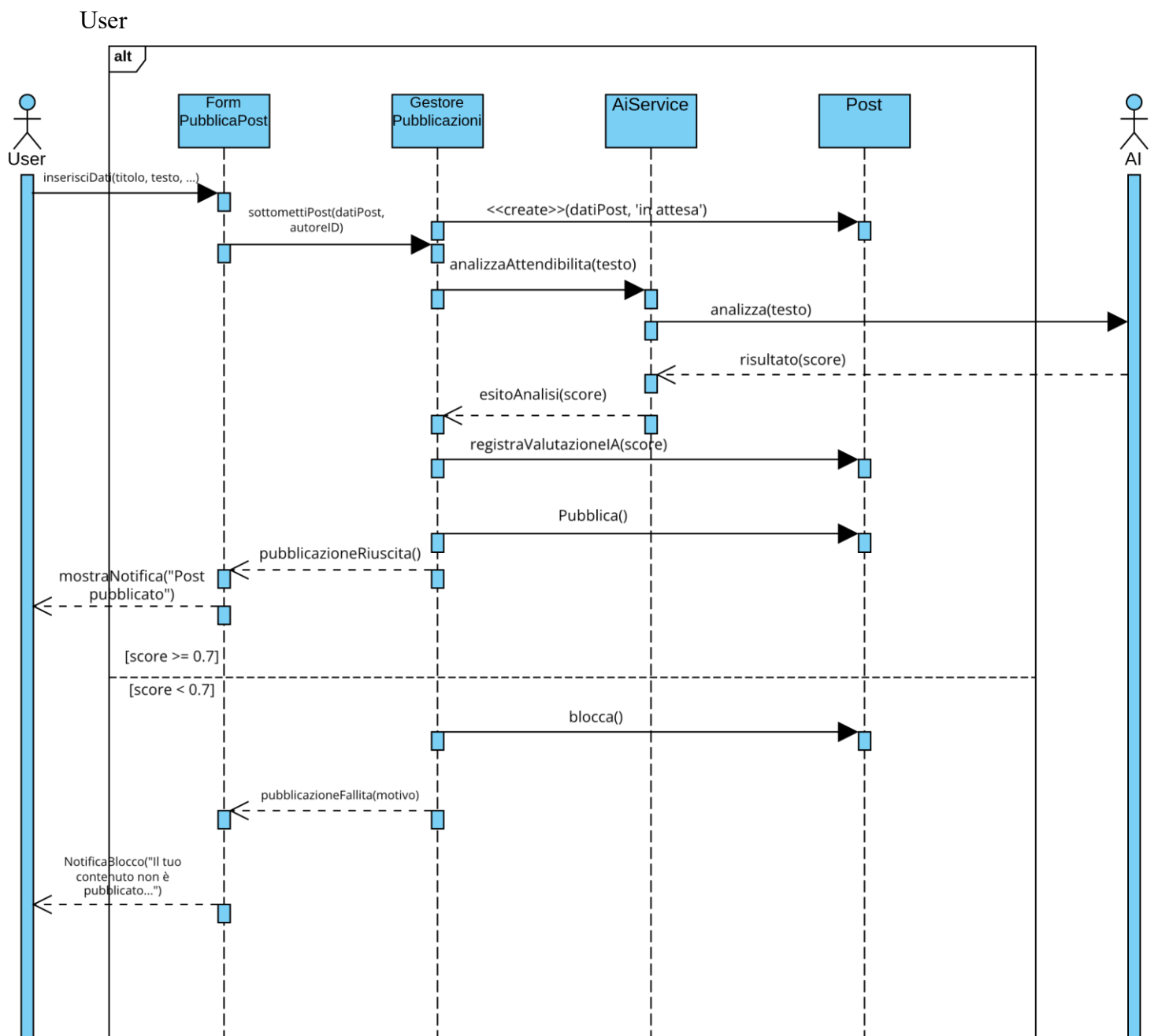
dalla GestorePubblicazioni e inviandolo al sistema AI per ottenere lo score di attendibilità.

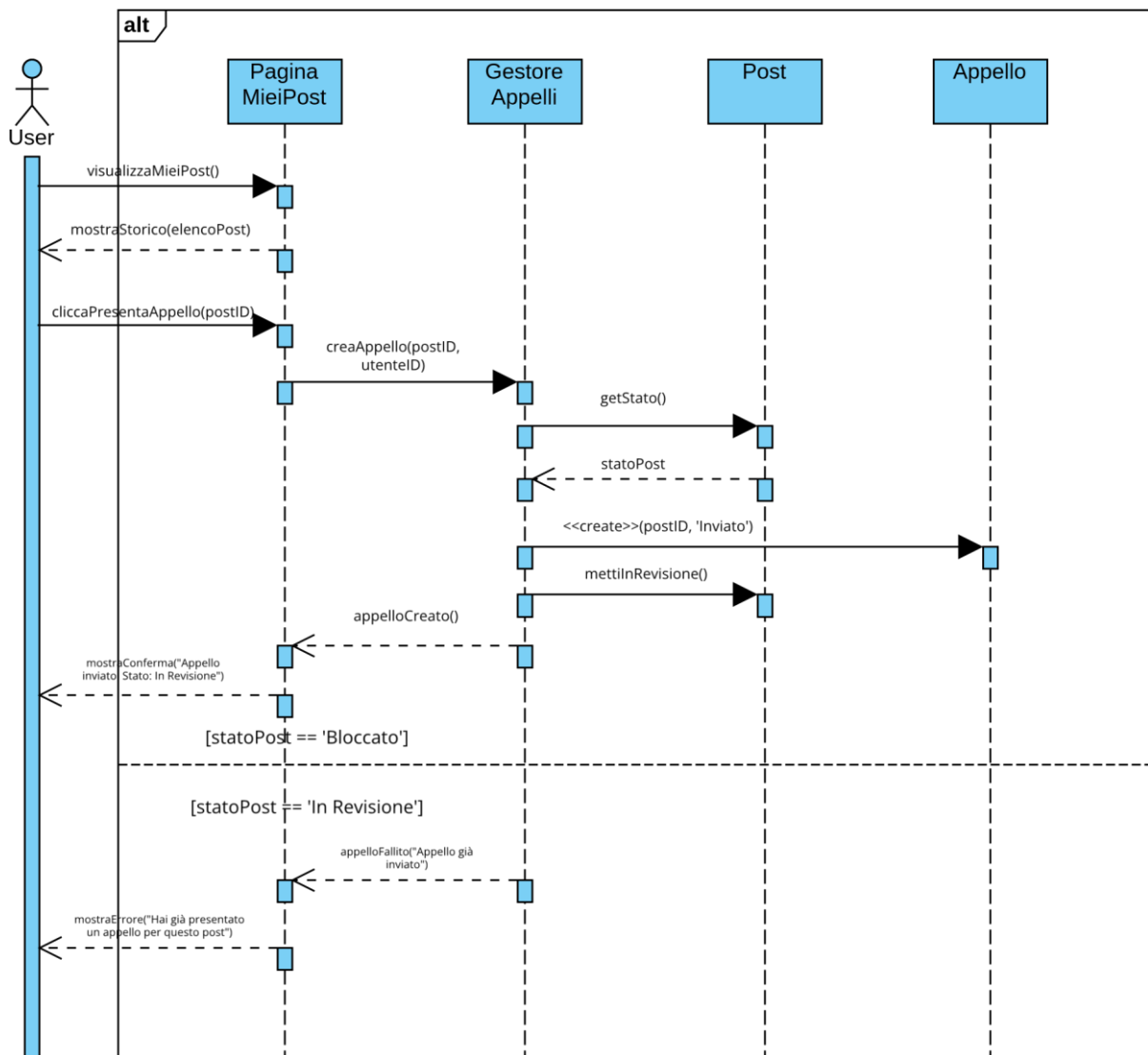


### 3.4.4. Dynamic Model

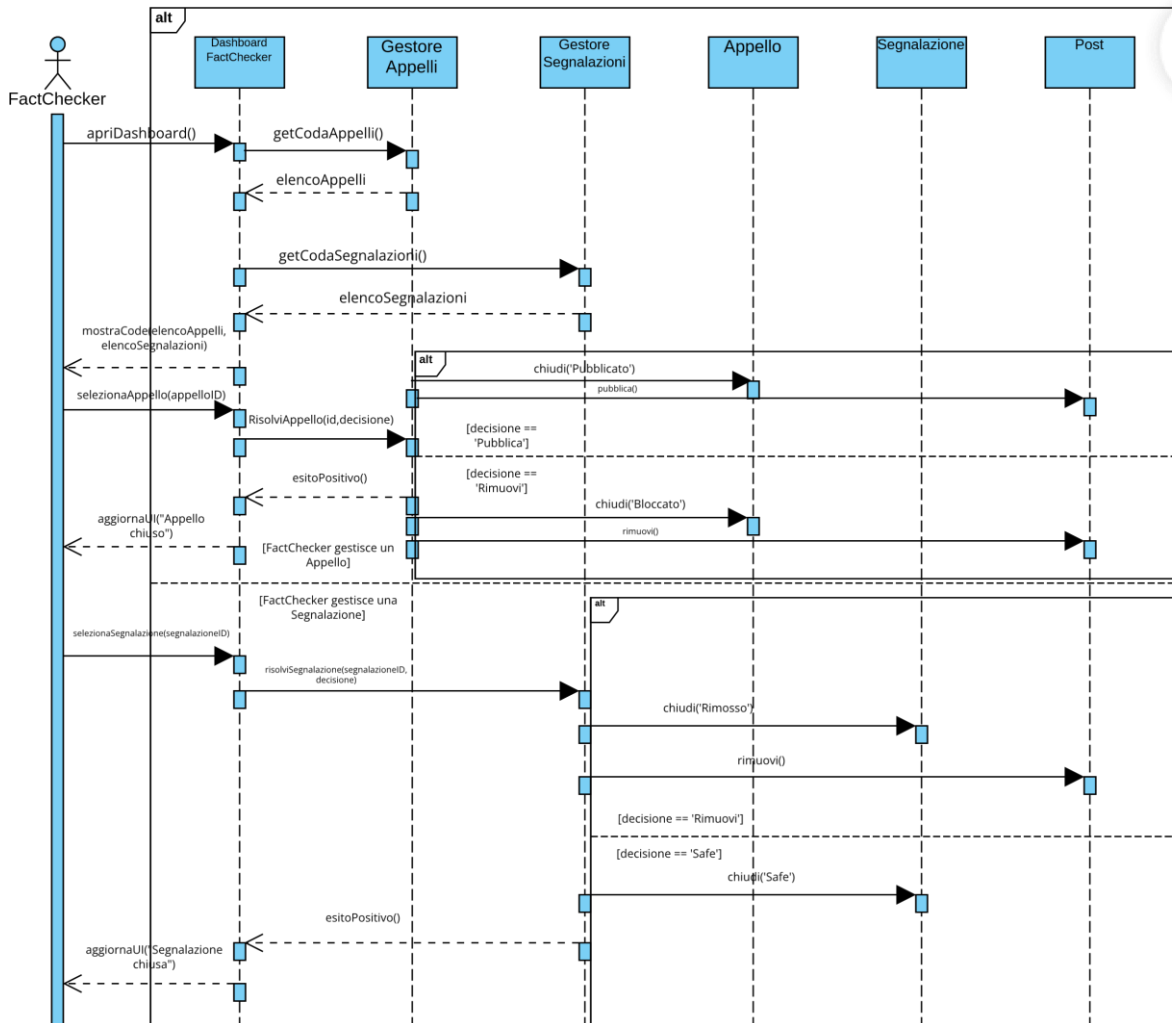
Guest





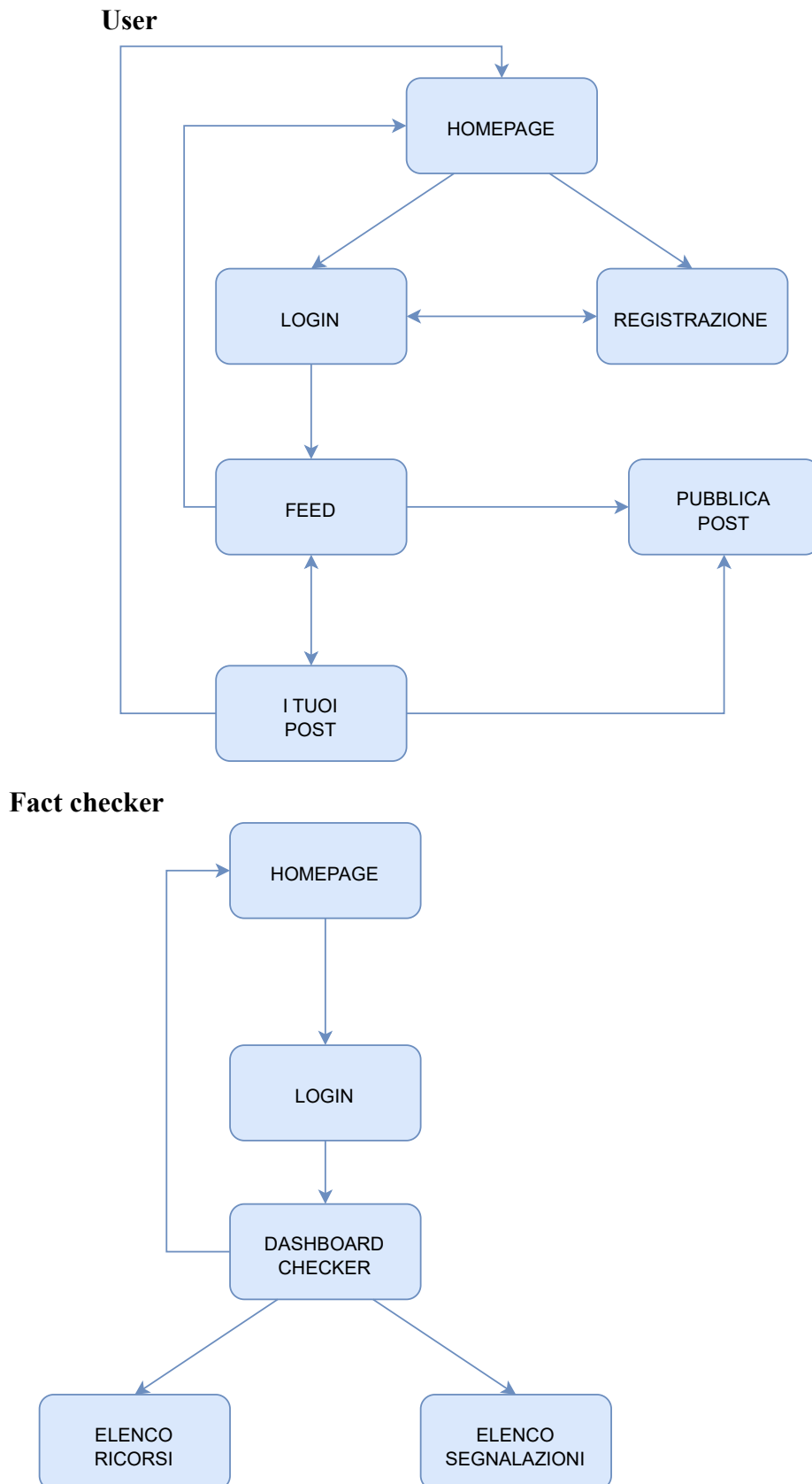


## Fact-checker



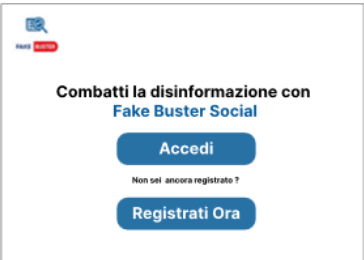
### 3.4.5. User interface-navigational paths and screen mock-ups

Navigation paths

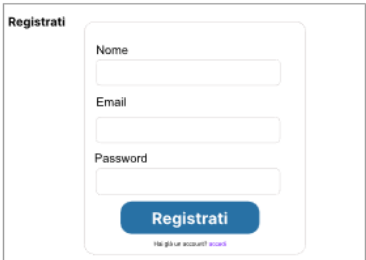


Mock-ups

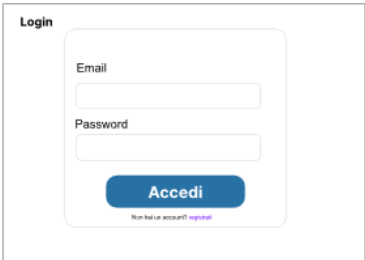
1



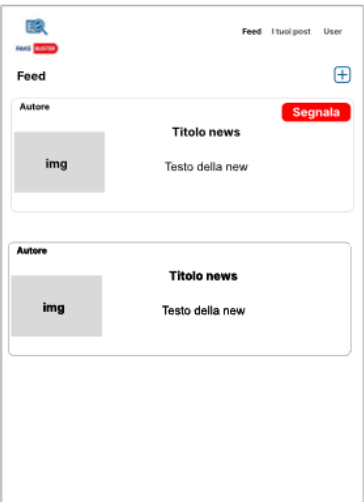
2



3



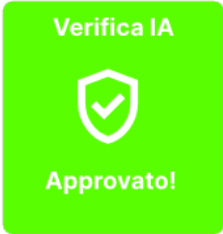
4



5



6



7



