PROBLEM STATEMENT

**Glossario:**

1. Problem domain

2) Scenarios

3)Requirements

3.1 Functional and Nonfunctional requirements

4) Target environment

5) Client Acceptance Criteria

1. Problem domain

La disinformazione online rappresenta una delle principali sfide del mondo digitale moderno. La facilità di condivisione dei contenuti e l’assenza di un controllo centralizzato rendono difficile distinguere tra informazione verificata e notizie false.

**FakeBuster** nasce come progetto accademico con lo scopo di:

* Supportare gli utenti nell’identificazione automatica di contenuti potenzialmente falsi.
* Fornire un ambiente che unisca metodi di *ingegneria del software* e *intelligenza artificiale applicata al linguaggio naturale (NLP)*.
* Creare una piattaforma modulare e scalabile, in grado di crescere nel tempo.

Il sistema si pone l’obiettivo di:

* Automatizzare la raccolta e l’analisi di articoli e post online.
* Classificare i contenuti secondo il grado di attendibilità.
* Permettere il miglioramento continuo del modello grazie all’intervento umano dei fact-checker.

2)Scenarios

**Scenario 1 – Utente Base (Lettore)**

Un utente inserisce un link o un testo di un articolo per valutarne l’attendibilità. Il sistema restituisce una stima (es. “Alta attendibilità”, “Dubbia”, “Probabile fake”) e memorizza la consultazione nello storico personale.

**Scenario 2 – Utente Verificatore (Fact-checker)**

Un utente registrato con permessi avanzati accede allo storico globale, verifica articoli classificati automaticamente e li etichetta manualmente come *Fake* o *Vero*. Queste etichette vengono utilizzate per migliorare il modello di machine learning.

**Scenario 3 – Sistema di apprendimento continuo**

Il sistema aggiorna periodicamente il modello di classificazione sfruttando i nuovi dati etichettati dai verificatori, migliorando nel tempo la precisione.

3)Requirements

3.1)Functional Requirements

1. **Crawling e Raccolta dati**
   * Il sistema deve essere in grado di raccogliere articoli e post da fonti online specificate.
   * Deve supportare il salvataggio dei metadati (fonte, data, autore, titolo, testo, ecc.).
2. **Analisi e Classificazione**
   * Analisi testuale tramite NLP (es. TF-IDF, embedding, modelli BERT di base).
   * Output: punteggio di affidabilità
3. **Interfaccia Utente (Frontend)**
   * Inserimento di link o testo per la valutazione.
   * Visualizzazione dei risultati in forma comprensibile (percentuale di attendibilità, grafico, storico personale).
4. **Gestione Utenti**
   * Registrazione e autenticazione (Base, Fact-checker).
   * Accesso differenziato alle funzionalità e ai dati.
5. **Database e Storico**
   * Archiviazione di articoli, risultati di classificazione, etichette manuali e storico degli utenti.
6. **Apprendimento Automatico e Feedback Loop**
   * Aggiornamento periodico del modello di classificazione basato sui dati etichettati.

3.2)Nonfunctional Requirements

1. **Usabilità**  
   L’interfaccia deve essere intuitiva, con un design chiaro e accessibile anche a utenti non tecnici.
2. **Scalabilità**  
   Il sistema deve poter gestire un numero crescente di articoli e utenti senza degrado significativo delle prestazioni.
3. **Affidabilità**  
   I dati devono essere salvati in modo sicuro e persistente; il sistema deve tollerare guasti parziali.
4. **Sicurezza**
   * Autenticazione sicura (token/sessione).
   * Gestione dei ruoli e delle autorizzazioni.
   * Protezione da input malevoli (iniezioni, spam, ecc.).
5. **Manutenibilità**  
   Codice modulare, documentato e gestito tramite versioning (GitHub).  
   Struttura pronta per estensioni future (es. supporto immagini/video).
6. **Compatibilità e Portabilità**  
   Funzionamento previsto su browser desktop e mobile moderni.

4)Target Environment

Poiché il progetto è in fase iniziale, le tecnologie esatte non sono ancora definitive. Tuttavia, la previsione è:

* **Ambiente di sviluppo(previsto):**
  + IDE: Visual Studio Code / PyCharm.
  + Versioning: GitHub.
  + Modellazione: UML con StarUML / PlantUML.
* **Stack tecnologico (previsto):**
  + Backend: Python (Flask / FastAPI) o Node.js.
  + Frontend: HTML/CSS
  + Database: MySQL.
  + AI/NLP: librerie Python (scikit-learn, spaCy, Transformers).