Ga

Bilkent University CS491

Analysis and Requirements Document

T2426 - CheckMate

Ayberk Berat Eroğlu - 22103675

Alp Batu Aksan - 22103246

Pelin Öner - 22102409

İpek Sönmez - 22103939

Efe Tuna Can - 22102127

1. Introduction

In a world overwhelmed by rapidly spreading misinformation, ensuring that what we read online is accurate and reliable has become a pressing challenge. Our project CheckMate is a browser extension that provides fact-checking and political bias detection services. Currently, most fact-checking is done by humans researching the topic. For example, Meta, the company behind Facebook and Instagram, uses human resources for fact-checking [1]. This approach is often too slow to keep up with the overwhelming flood of information. This is because fact-checking is a long process and requires serious human resources for each article to be fact-checked [2]. Our project CheckMate aims to automate fact-checking with its Artificial Intelligence (AI) driven system and ultimately provide a quick and easy way to combat misinformation online. Slowly, the internet is becoming the primary source of news people see; in some countries, such as the UK, it is the biggest news source [3]. Thus, CheckMate will work with all major news and social media websites to provide its services.

For its services, CheckMate will evaluate each article and assign a reliability score for them. To do this, natural language processing (NLP) will be used to determine whether or not the language used is biased. It will analyze the article for any political bias and will use this information for its reliability score. Moreover, it will check other sources on the internet for confirmatory or contradictory information regarding the information in the selected news article through search Application Programming Interfaces (APIs). CheckMate will also use the metadata of the news article, if possible, such as the website on which it was published or its author. Using

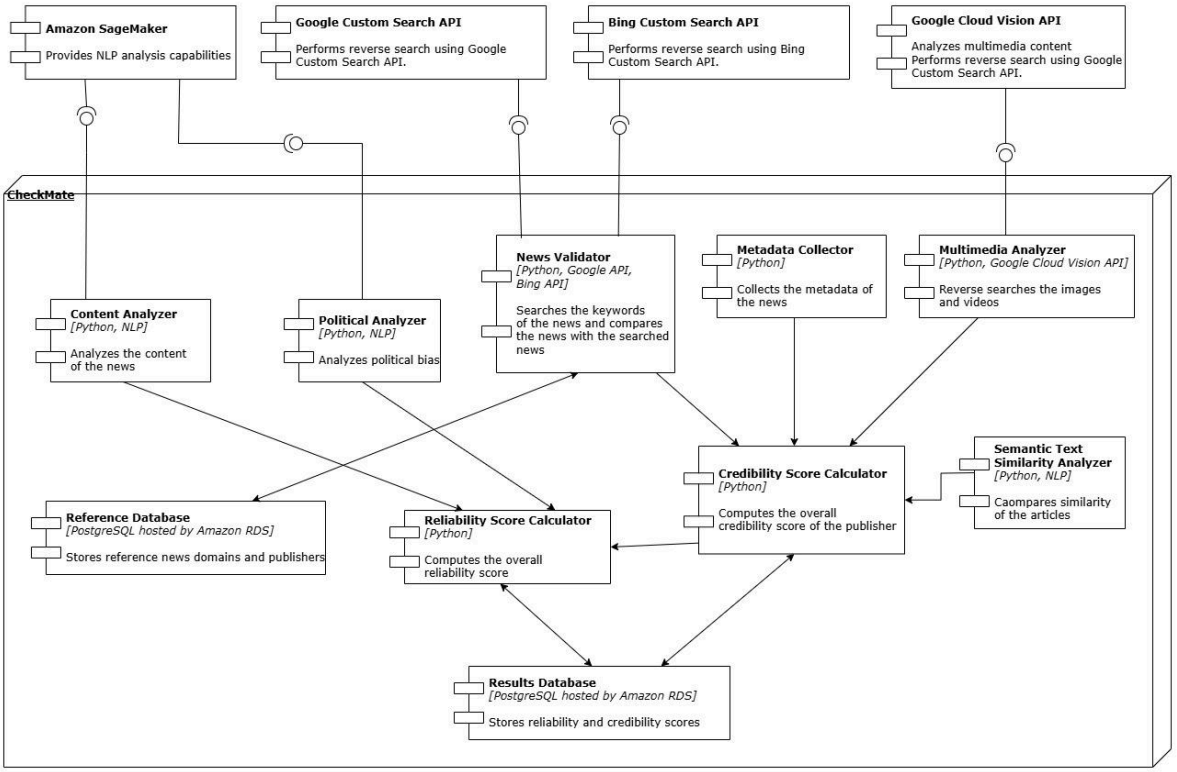
the metadata of the news article CheckMate will generate a credibility score for the news' source as well regarding whether the source released fake news before, has political inclination, uses biased language, etc. This credibility score will contribute to the overall reliability score of the news article. Additionally, CheckMate will perform reverse searches on visuals, such as photos, thumbnails, and videos, used in news articles to check for inconsistencies or mismatched contexts through search APIs. This reverse search will also provide us with whether or not those visuals were used before in different contexts. After CheckMate completes its analysis, the system will show the reliability score of the article alongside the reasons for its score. These reasons will range from having a high political bias to providing contradictory information to other trustworthy sources.

2. Current System

The following changes were made:

- A new model was added to compare the similarity of the articles.

New high level system diagram



3. Proposed System

3.1. Overview

Checkmate is a tool designed to analyze the credibility, context, and political bias of news articles by leveraging advanced AI models and a robust backend architecture. Users can submit articles for evaluation, after which the application processes the content through multiple analytical components. The

language analyzer model assesses the grammar and linguistic structures of the article, the political spectrum analyzer model classifies the article along the political spectrum, and the image verification module performs reverse searches on images within the article to validate their context and authenticity. The backend aggregates these results to calculate a credibility score and generates a detailed report outlining the reasoning behind the evaluation. Additionally, users can compare articles with similar content and access previously analyzed reports for further reference.

The application also includes comprehensive account management and customization features. Users can create accounts, log in, and manage their profiles by changing passwords, updating payment plans, or upgrading to premium. For those using browser extensions, the application allows settings to be modified to enhance functionality. The application ensures data privacy and provides mechanisms to handle common issues, such as login or signup errors. Feedback from users is received and stored to improve the overall system performance.

With these capabilities, Checkmate provides an efficient and systematic approach to evaluating news articles. Its integration of AI-driven analysis ensures that the credibility and bias of content are assessed with precision, enabling users to critically engage with information. The combination of technical rigor and user-oriented design makes Checkmate a valuable resource for fostering media literacy and informed decision-making.

3.2. Functional Requirements

- The system will maintain a reference database of known news websites and social media platforms commonly used for journalism, as demonstrated in [4], which will be regularly updated to incorporate new and emerging platforms.
- The system will compare the Uniform Resource Locators (URLs) of each website that the user enters with that reference database.
- The system will collect metadata (e.g., publication date, location, and author) of the news article on the new source.
- The system will validate the time and place of publication against credible sources to verify accuracy.
- The system will update the reliability score of news websites periodically based on newly available information.
- The system will analyze the publisher of each news article to check whether they are deemed reliable.
- The system will allow users to access news domains that have published false information to update the publisher's reliability.
- The system will analyze the content of each news article to assess its factual accuracy by performing reverse searches using APIs such as Google Custom Search API [5], Bing News Search API [6], and Bing Custom Search API [7].
- The system will use an NLP model to determine whether the article is biased or not.
- The system will flag statements with no credible references or with conflicting information.

- The system will analyze the language and tone of political news articles to determine political polarity using an NLP model.
- Based on sentiment analysis, the system will classify political slants along a compass (e.g., left-leaning, right-leaning, centrist).
- The system will display the political slant of each article on a visual compass, enabling users to see the degree of bias.
- The system will check if the news article has appeared on other reliable news websites.
- The system will compare the information across multiple sources to detect discrepancies and increase the accuracy score if verified by multiple independent sources.
- The system will lower the accuracy score if conflicting information is found in other credible sources.
- The system will detect if images or videos are present in the news article.
- The system will analyze multimedia content to identify instances of use in unrelated contexts by utilizing tools such as Google Cloud Vision API's label detection feature [8].
- The system will verify the relevance and accuracy of multimedia content by comparing it against known sources through reverse search methods using Google Cloud Vision's web detection feature [9] and Bing Image Search API [10].
- The system will compute an overall accuracy score for each news article based on the metadata, source reliability, publisher credibility, content accuracy, political slant, cross-verification, and multimedia verification.

- The system will display the accuracy score to users, showing a breakdown of how each criterion contributes to the final score.
- The system will allow users to submit feedback if they believe a news article is inaccurate or biased.
- The system will provide a visual interface to show each news article's accuracy score and political compass.
- The system will allow users to view detailed information for each accuracy criterion, including explanations and sources used in the analysis.
- The system will allow administrators to update the databases of publishers and websites with new information as it becomes available.

3.3. Nonfunctional Requirements

3.3.1. Usability

- The system will provide an intuitive and user-friendly interface, allowing users to easily access each article's reliability score and political compass.
- The system will display clear visual indicators (e.g., color-coded scores, compass graphics) to effectively convey the reliability and political slant of articles.
- The system will allow users to provide feedback on accuracy and bias with minimal steps.
- The system will be accessible on major Chromium browsers (e.g., Chrome, Opera, Edge) and optimized for responsive display across different screen sizes [11].

3.3.2. Reliability

- The system will ensure accuracy and consistency in reliability scores and political spectrum classification by regularly updating the reference databases and verifying data sources.
- The system will have mechanisms to flag and handle inconsistencies or errors in analysis.

3.3.3. Performance

- The system will perform efficient data processing, using caching and optimized algorithms to minimize latency in news verification and NLP model usage.
- The system will handle high traffic loads by distributing requests and limiting non-essential checks when necessary to maintain a seamless user experience.

3.3.4. Supportability

- The system will be easy to update to accommodate changes in the reference databases, external APIs, or NLP models.
- The system's codebase will be modular and well-documented, enabling maintainers to make modifications or add new functionalities with minimal risk of disruption.
- The system will log significant events and errors, allowing developers to diagnose issues and track the system's health and performance.

3.3.5. Scalability

- The system will be able to scale to accommodate an increasing number of users and a higher frequency of requests without impacting performance.
- The system's architecture will allow for integration with additional NLP models and new APIs for extended capabilities (e.g., enhanced multimedia analysis or additional political slant dimensions).
- The system will support future expansion of reference databases, maintaining efficient operations as the number of sources and articles grows.

3.4. Pseudo Requirements

Checkmate is a browser extension based on the Chromium framework, which ensures compatibility with Chromium-based browsers, including but not limited to Google Chrome and Opera [11].

3.5. System Models

3.5.1. Scenarios

3.5.1.1 Sign Up

- **Use-case Name**: Sign up
- **Actor**: User
- **Entry Condition**: User has opened the browser extension
- **Exit Condition**: User successfully created an account

- **Flow of Events**:
    1. User clicks on "Sign Up" button
    2. User enters email address
    3. User creates a password
    4. System validates email and password
    5. System sends verification email
    6. User confirms email through verification link
    7. Account is created successfully

3.5.1.2 Sign Up Error

- **Use-case Name**: Handle Sign Up Errors
- **Actor**: User
- **Entry Condition**: User is attempting to create an account
- **Exit Condition**: User is informed of registration issues
- **Alternative Flows**:
    1. Invalid Email Address
        - System displays error message
        - User is prompted to enter a valid email
    2. Password Ineligibility
        - System checks password strength
        - Displays specific password requirements
    3. Verification Email Failure
        - System fails to send verification email
        - User can request resend verification link

3.5.1.3 Login

- **Use-case Name**: Login
- **Actor**: User
- **Entry Condition**: User has registered account
- **Exit Condition**: User successfully logs into the system
- **Flow of Events**:
    1. User enters email address
    2. User enters password
    3. System validates credentials
    4. User is logged into the extension

3.5.1.4 Login Error

- **Use-case Name**: Handle Login Errors
- **Actor**: User
- **Entry Condition**: User is attempting to log in
- **Exit Condition**: User is informed of login issues
- **Alternative Flows**:
    1. Invalid Email Address
        - System displays error message
        - Prompts user to check email input
    2. Incorrect Password
        - System shows password error
        - Offers password reset option
    3. Session Start Failure

        - System provides technical error information
        - Suggests user try again or contact support

3.5.1.5 Analyze News Article

- **Use-case Name**: Analyze News Article
- **Actor**: User
- **Entry Condition**: User is logged in and has selected an article
- **Exit Condition**: Comprehensive article analysis is displayed
- **Flow of Events**:
    1. User navigates to news article
    2. User activates browser extension
    3. Extension captures article content
    4. Backend initiates analysis process
        - Reverse image search
        - Language analysis
        - Political spectrum classification
        - Web article comparison
    5. System generates credibility score
    6. Detailed analysis report is presented to user

3.5.1.6 Browse Past Analyses

- **Use-case Name**: Browse Past Analyses
- **Actor**: User
- **Entry Condition**: User is logged in

- **Exit Condition**: User views previous article analyses
- **Flow of Events**:
    1. User selects "Past Searches" option
    2. System retrieves saved article analyses
    3. User can view detailed reports of previous analyses

3.5.1.7 Change Password

- **Use-case Name**: Change Password
- **Actor**: User
- **Entry Condition**: User is logged in
- **Exit Condition**: Password is successfully changed
- **Flow of Events**:
    1. User navigates to account settings
    2. User selects "Change Password"
    3. User enters current password
    4. User enters new password
    5. User confirms new password
    6. System validates password change conditions

3.5.1.8 Password Change Error

- **Use-case Name**: Handle Password Change Errors
- **Actor**: User
- **Entry Condition**: User is attempting to change password
- **Exit Condition**: User is informed of password change issues

- **Alternative Flows**:
    1. New Password Same as Old Password
        - System blocks password change
        - Prompts user to choose a different password
    2. Password Confirmation Mismatch
        - System shows error message
        - Asks user to re-enter new password
    3. Password Reset Email Issues
        - System fails to send reset email
        - Provides alternative reset methods

3.5.1.9 Modify Extension Settings

- **Use-case Name**: Modify Extension Settings
- **Actor**: User
- **Entry Condition**: User is logged in
- **Exit Condition**: Extension settings are updated
- **Flow of Events**:
    1. User accesses extension settings
    2. User can modify:
        - Display preferences
        - Notification settings
    3. User saves new configuration

3.5.1.10 Reverse Image Search

- **Use-case Name:** Reverse Search Images in the Article
- **Actor:** Backend
- **Entry Condition:** User submits an article for analysis.
- **Exit Condition:** Reverse search results for the images are obtained.
- **Flow of Events:**
    1. System extracts images from the submitted article.
    2. For each image, the system performs a reverse search.
    3. The reverse search results are retrieved and logged.

3.5.1.11 Send to Language Analyzer

- **Use-case Name:** Send Article to Language Analyzer Model
- **Actor:** Backend
- **Entry Condition:** Article text is available for analysis.
- **Exit Condition:** Language analysis results are generated.
- **Flow of Events:**
    1. System sends the article text to the language analyzer model.
    2. The model processes the text and identifies language properties (e.g., tone, sentiment, complexity).
    3. The system receives and logs the analysis results.

3.5.1.12 Send to Political Analyzer

- **Use-case Name:** Send Article to Political Spectrum Analyzer Model
- **Actor:** Backend
- **Entry Condition:** Article text is available for analysis.

- **Exit Condition:** Political spectrum analysis results are generated.
- **Flow of Events:**
    1. System sends the article text to the political spectrum analyzer model.
    2. The model evaluates the text for political bias and stance.
    3. The system retrieves and logs the results.

3.5.1.13 Search Similar Articles

- **Use-case Name:** Search Similar Articles on the Web
- **Actor:** Backend
- **Entry Condition:** User submits an article for analysis.
- **Exit Condition:** List of similar articles is generated.
- **Flow of Events:**
    1. System extracts key points from the article for comparison.
    2. A web search is initiated using these key points.
    3. The system collects articles that closely match the submitted article.
    4. The results are ranked based on relevance and logged for comparison.

3.5.1.14 Compare Similar Articles with the Article

- **Use-case Name:** Compare Article Contents with Search Results
- **Actor:** Backend
- **Entry Condition:** Article text and search results are available.

- **Exit Condition:** Content comparison analysis is completed.
- **Flow of Events:**
    1. System performs keyword and phrase matching between the article and similar articles.
    2. Contextual consistency and similarity scores are calculated.
    3. Major differences in content or claims are flagged.
    4. Results are logged for inclusion in the credibility report.

3.5.1.15 Calculate Credibility Score

- **Use-case Name:** Calculate Credibility Score of the Article
- **Actor:** Backend
- **Entry Condition:** All required analysis data is available.
- **Exit Condition:** Credibility score is calculated and logged.
- **Flow of Events:**
    1. System aggregates all analysis results (language, political spectrum, reverse search, content comparison).
    2. Each component is assigned a weighted contribution to the overall score.
    3. The credibility score is calculated using a predefined formula.
    4. The final score is logged and added to the report.

3.5.1.16 Create Report

- **Use-case Name:** Create a Report for the Credibility Score
- **Actor:** Backend

- **Entry Condition:** Credibility score and analysis data are available.
- **Exit Condition:** Detailed report is generated.
- **Flow of Events:**
    1. System collects all analysis data and the calculated credibility score.
    2. A structured report is generated, summarizing the findings.
    3. Highlights include discrepancies, key points, and the final score.
    4. The report is stored for future reference and sent to the user.

3.5.1.17 Send Information Back

- **Use-case Name:** Send Information Back to the User
- **Actor:** Backend
- **Entry Condition:** All analysis results are available.
- **Exit Condition:** Results are successfully sent to the user.
- **Flow of Events:**
    1. System formats the analysis results for presentation.
    2. Results are sent to the user interface for display.
    3. User receives a notification or sees the results on their dashboard.

3.5.1.18 Maintain a Database

- **Use-case Name:** Maintain a Database of Previously Analyzed Articles
- **Actor:** Backend

- **Entry Condition:** New analysis results are available.
- **Exit Condition:** Results are successfully stored in the database.
- **Flow of Events:**
    1. System checks if the article has been previously analyzed.
    2. If not, the analysis results are added to the database.
    3. The database is updated with the user's feedback (if provided).
    4. System logs the update for future queries.

3.5.1.19 Save to Past Searches

- **Use-case Name:** Save the Result User's Past Searches
- **Actor:** Backend
- **Entry Condition:** User initiates a search.
- **Exit Condition:** Search data is logged in the user's history.
- **Flow of Events:**
    1. System records the URL or content of the user's search.
    2. Metadata (e.g., time, source) is attached to the search data.
    3. The search is saved in the user's history for future reference.

3.5.1.20 Receive and Store User Feedback

- **Use-case Name:** Receive and Store User Feedback
- **Actor:** Backend
-
- **Entry Condition:** User submits feedback.
- **Exit Condition:** Feedback is logged successfully.

- **Flow of Events:**
    1. User provides feedback through the app interface.
    2. System validates and logs the feedback.
    3. Feedback is associated with the relevant analysis or system feature.
    4. The data is stored in the database for future improvements.

3.5.1.21 Analyze Article

- **Use-case Name:** Analyze the Given Article
- **Actor:** Political Spectrum Analyzer Model
- **Entry Condition:** Article text is sent to the model.
- **Exit Condition:** Article analysis is completed and results are returned.
- **Flow of Events:**
    1. The model receives the article text from the backend.
    2. The text is processed to identify political keywords, phrases, and sentiments.
    3. The model performs a detailed analysis to determine political alignments and perspectives.
    4. The analysis results are sent back to the backend.

3.5.1.22 Classify Article on Political Spectrum

- **Use-case Name:** Classify the Given Article on the Political Spectrum
- **Actor:** Political Spectrum Analyzer Model
- **Entry Condition:** Processed article data is available.

- **Exit Condition:** Political spectrum classification is completed and returned.
- **Flow of Events:**
    1. The model uses predefined political metrics to classify the article.
    2. The article is assigned a position on the political spectrum (e.g., left, center, right).
    3. The classification result is sent back to the backend.

3.5.1.23 Return Political Analysis to Backend

- **Use-case Name:** Return the Result Back to the Backend
- **Actor:** Political Spectrum Analyzer Model
- **Entry Condition:** Analysis and classification are complete.
- **Exit Condition:** Results are successfully sent to the backend.
- **Flow of Events:**
    1. The model packages the analysis and classification data into a structured format.
    2. The results are sent back to the backend via the API.
    3. The backend confirms receipt of the data.

3.5.1.24 Receive and Store User Feedback

- **Use-case Name:** Analyze the Grammar of the Given Article
- **Actor:** Language Analyzer Model
- **Entry Condition:** Article text is sent to the model.

- **Exit Condition:** Grammar analysis is completed, and results are returned.
- **Flow of Events:**
    1. The model receives the article text from the backend.
    2. The text is processed to identify grammatical structures and rules.
    3. Any grammatical mistakes are detected and logged.
    4. The results are sent back to the backend.

3.5.1.25 Find Article Mistakes

- **Use-case Name:** Find the Mistakes in the Article

  Actor: Language Analyzer Model
- **Entry Condition:** Article text is processed for grammar analysis.
- **Exit Condition:** Grammar mistakes are identified and returned.
- **Flow of Events:**
    1. The model identifies spelling, punctuation, and syntax errors in the text.
    2. Mistakes are categorized (e.g., critical, minor).
    3. A report of mistakes is generated.
    4. The report is sent back to the backend.

3.5.1.26 Classify Article Based on Language Quality

- **Use-case Name:** Classify the Article's Language Quality
- **Actor:** Language Analyzer Model

- **Entry Condition:** Language analysis is completed.
- **Exit Condition:** Article is classified based on language quality.
- **Flow of Events:**
    1. The model evaluates the text based on grammar, vocabulary, and structure.
    2. The article is rated as **professional**, **good**, **mediocre**, **unprofessional**, or **bad**.
    3. The classification is logged for further processing.
    4. Results are sent back to the backend.

3.5.1.27 Return Language Results to Backend

- **Use-case Name:** Return the Results Back to the Backend
- **Actor:** Language Analyzer Model
- **Entry Condition:** Grammar and language quality analysis are completed.
- **Exit Condition:** Results are successfully sent to the backend.
- **Flow of Events:**
    1. The model compiles the analysis results into a structured report.
    2. The report is sent to the backend via the API.
    3. The backend confirms receipt of the data.

3.5.1.28 Open Extension

- **Use-case Name:** Open Extension on Browser
- **Actor:** User

- **Entry Condition:** User has the browser open with the extension installed.
- **Exit Condition:** Extension is launched successfully.
- **Flow of Events:**
    1. User clicks on the extension icon in the browser.
    2. The system initializes and displays the extension interface.

3.5.1.29 Select Article

- **Use-case Name:** Select News Article Website to Analyze
- **Actor:** User
- **Entry Condition:** User is on a news article website.
- **Exit Condition:** Analysis is initiated for the selected website.
- **Flow of Events:**
    1. User clicks the "Analyze Current Page" button.
    2. The system sends the current website's URL to the backend for processing.
    3. The system displays the analysis results to the user.

3.5.1.30 Change Plan

- **Use-case Name:** Change Current Plan
- **Actor:** User
- **Entry Condition:** User is logged in and has access to subscription settings.
- **Exit Condition:** Current plan is updated successfully.

- **Flow of Events:**
    1. User navigates to the “Upgrade” page
    2. User selects a new plan from the available options.
    3. The system updates the user's plan in the database.
    4. The system confirms the plan change to the user.

3.5.1.31 View Past Searches

- **Use-case Name:** View Past Searches
- **Actor:** User
- **Entry Condition:** User is logged in and selects the "Past Searches" option.
- **Exit Condition:** User views previously analyzed articles.
- **Flow of Events:**
    1. User navigates to the "Past Searches" section.
    2. The system retrieves saved analyses from the database.
    3. User views the list of past analyses and selects a detailed report if needed.

3.5.1.32 Edit Profile

- **Use-case Name:** Edit Profile
- **Actor:** User
- **Entry Condition:** User is logged in and selects the "Edit Profile" option.
- **Exit Condition:** Profile information is updated successfully.

- **Flow of Events:**
    1. User navigates to the "Edit Profile" section.
    2. User modifies their personal details (e.g., name, email).
    3. The system validates the input and updates the profile in the database.
    4. The system confirms the changes to the user.

3.5.1.33 View Analysis

- **Use-case Name:** View Analysis
- **Actor:** User
- **Entry Condition:** User initiates an analysis or selects a past analysis.
- **Exit Condition:** Analysis details are displayed to the user.
- **Flow of Events:**
    1. User clicks on a specific analysis result.
    2. The system retrieves the analysis report from the backend.
    3. The system displays the analysis details in a readable format.

3.5.1.34 Change Password

- **Use-case Name:** Change Password
- **Actor:** Use
- **Entry Condition:** User is logged in and selects the "Change Password" option.
- **Exit Condition:** Password is updated successfully.
- **Flow of Events:**

1. User navigates to the "Change Password" section.
2. User enters the current password and a new password.
3. The system validates the current password and confirms the new password match.
4. The system updates the password in the database and notifies the user of success.

3.5.1.35 Log Out

- **Use-case Name:** Log Out
- **Actor:** User
- **Entry Condition:** User is logged in and selects the "Log Out" option.
- **Exit Condition:** User is logged out of their account.
- **Flow of Events:**
    1. User clicks the "Log Out" button.
    2. The system ends the user's session and redirects them to the login page.

3.5.1.36 Settings

- **Use-case Name:** Change Settings
- **Actor:** User
- **Entry Condition:** User is logged in and selects the "Settings" option.
- **Exit Condition:** Settings are updated successfully.
- **Flow of Events:**
    1. User navigates to the "Settings" section.

2. User modifies desired settings (e.g., notification preferences).
3. The system saves the updated settings in the database.
4. The system confirms the changes to the user.

3.5.1.37 View Detailed Analysis

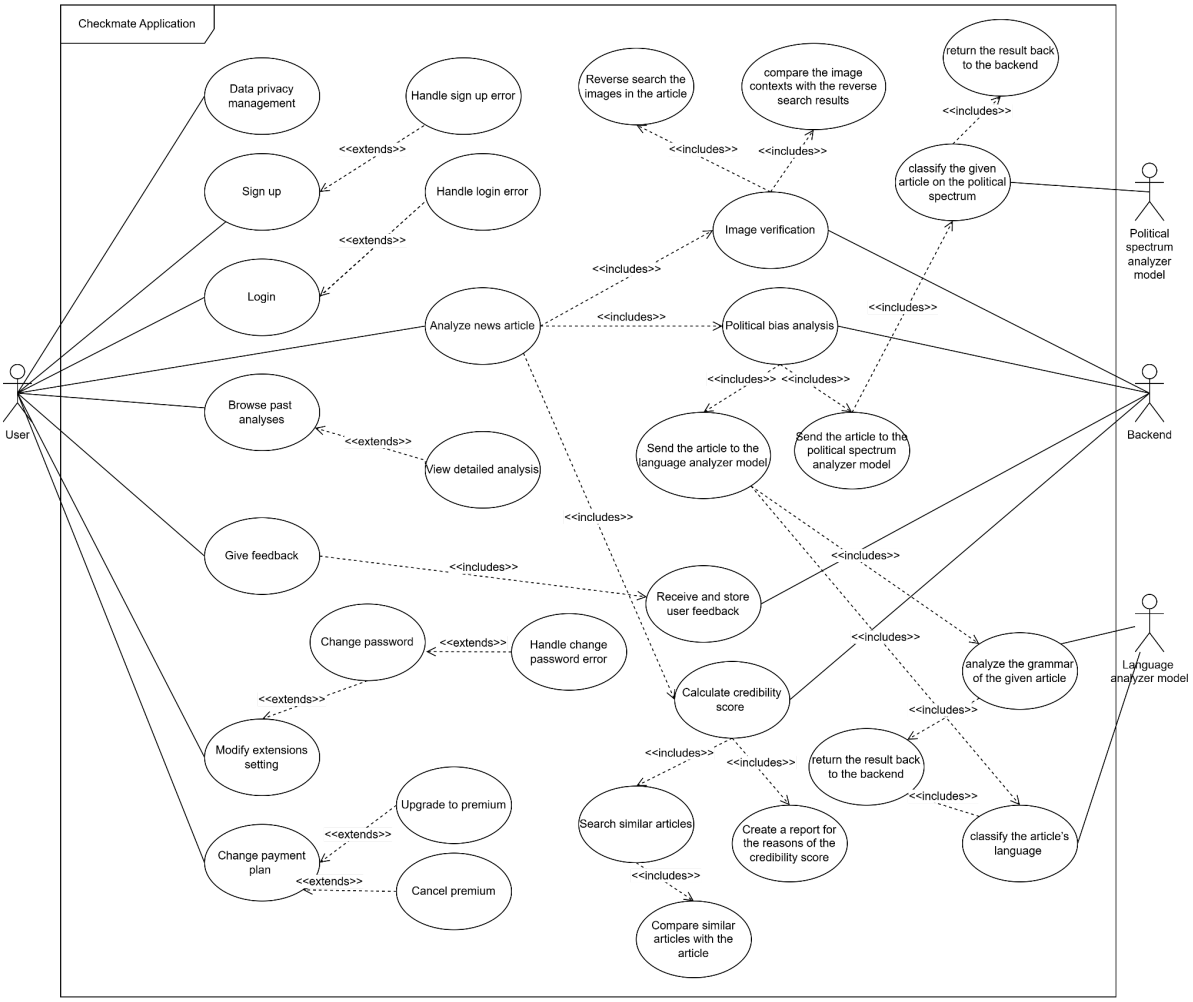
- **Use-case Name:** View Detailed Analysis
- **Actor:** User
- **Entry Condition:** User selects a detailed analysis option.
- **Exit Condition:** A detailed analysis report is displayed.
- **Flow of Events:**
    1. User clicks on "View Detailed Analysis" for a specific report.
    2. The system retrieves additional data and statistics for the analysis.
    3. The system displays a comprehensive report to the user.

3.5.1.38 Report Mistake

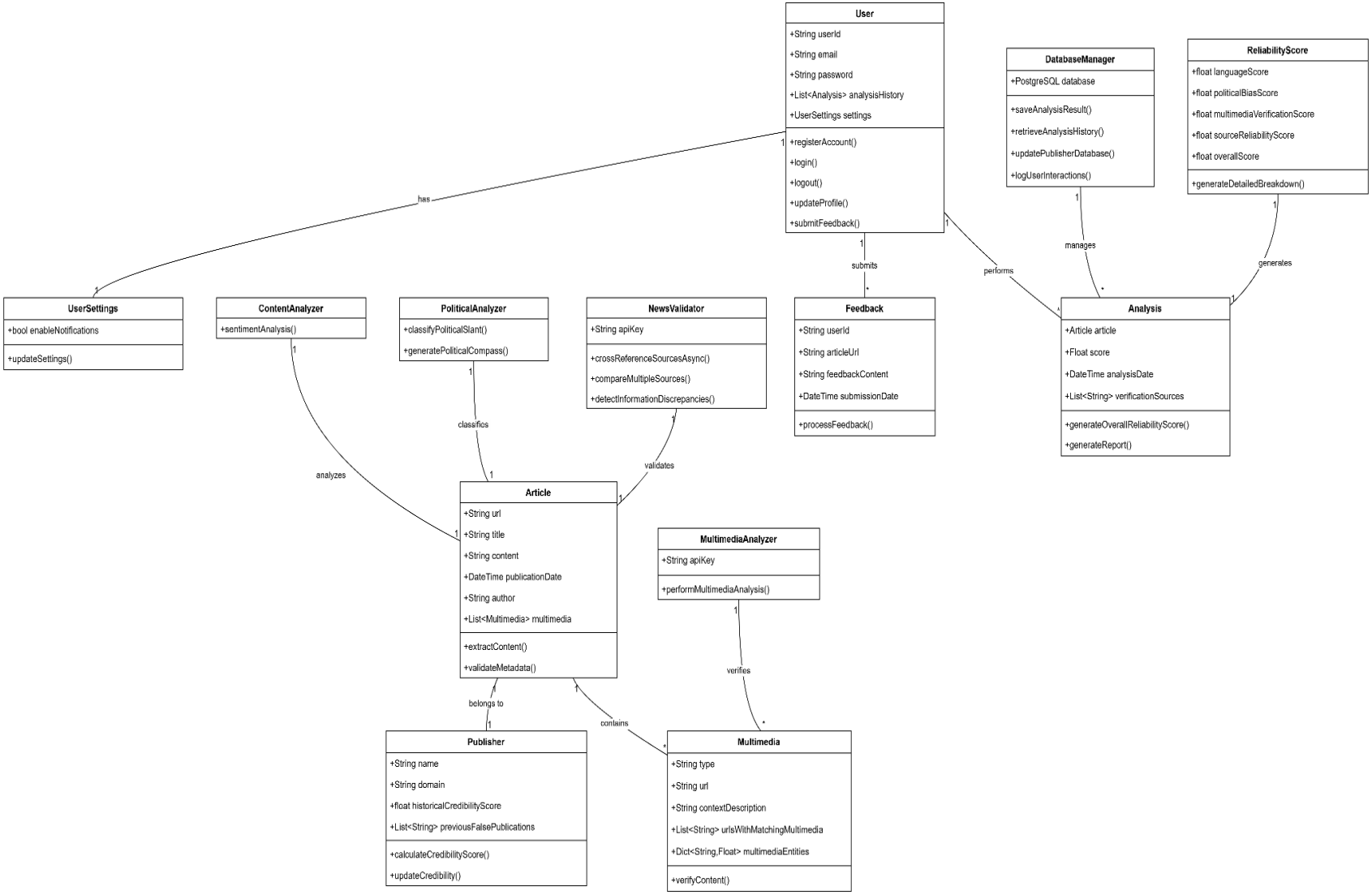
- **Use-case Name:** Report Mistake
- **Actor:** User
- **Entry Condition:** User notices an issue with an analysis report.
- **Exit Condition:** The mistake is reported to the system successfully.
- **Flow of Events:**
    1. User selects the "Report Mistake" option for a specific analysis.
    2. User fills out a form describing the issue.
    3. The system validates the report and logs it in the database.

4. The system confirms the report submission to the user.

3.5.2. Use Case Model

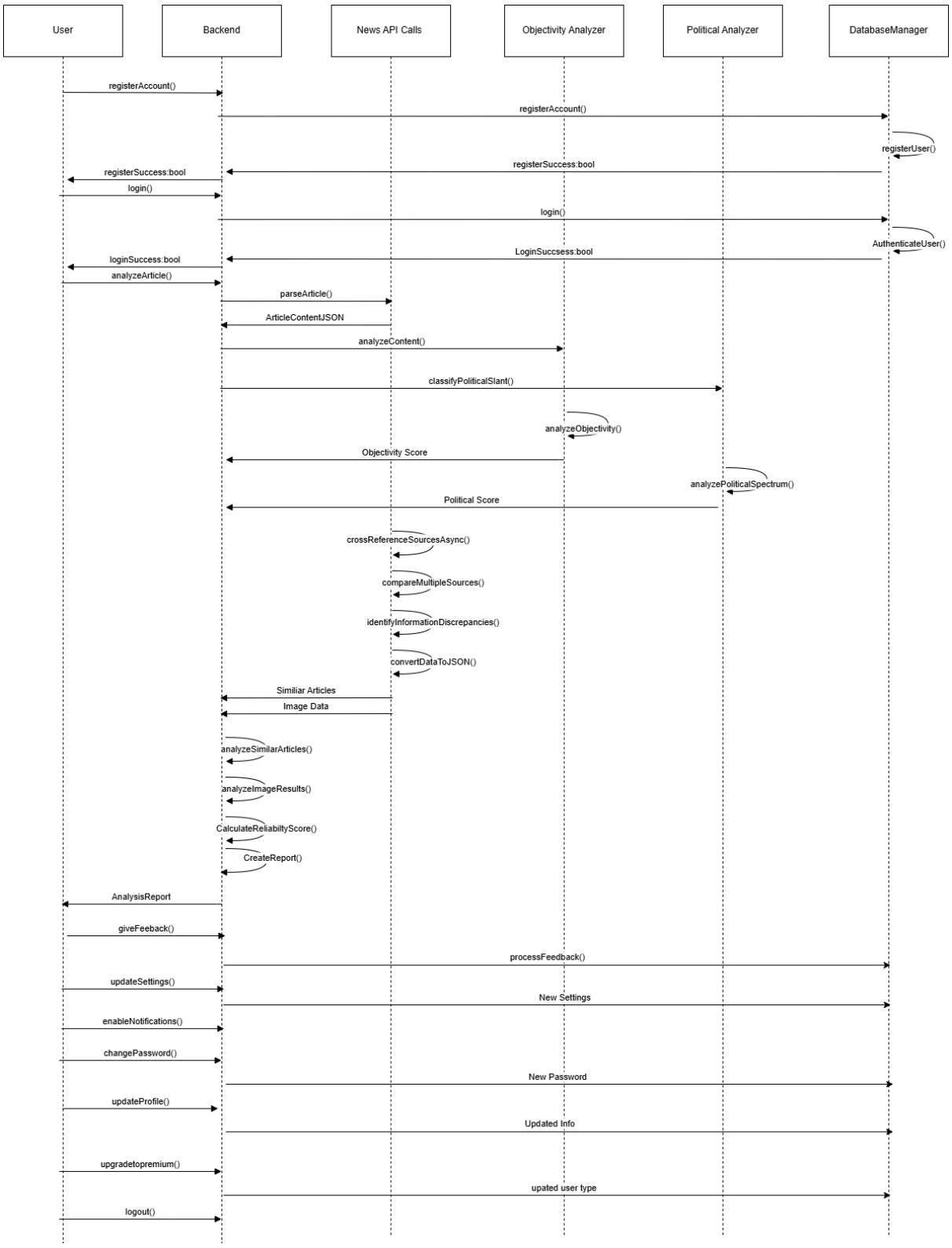


3.5.3. Object and Class Model



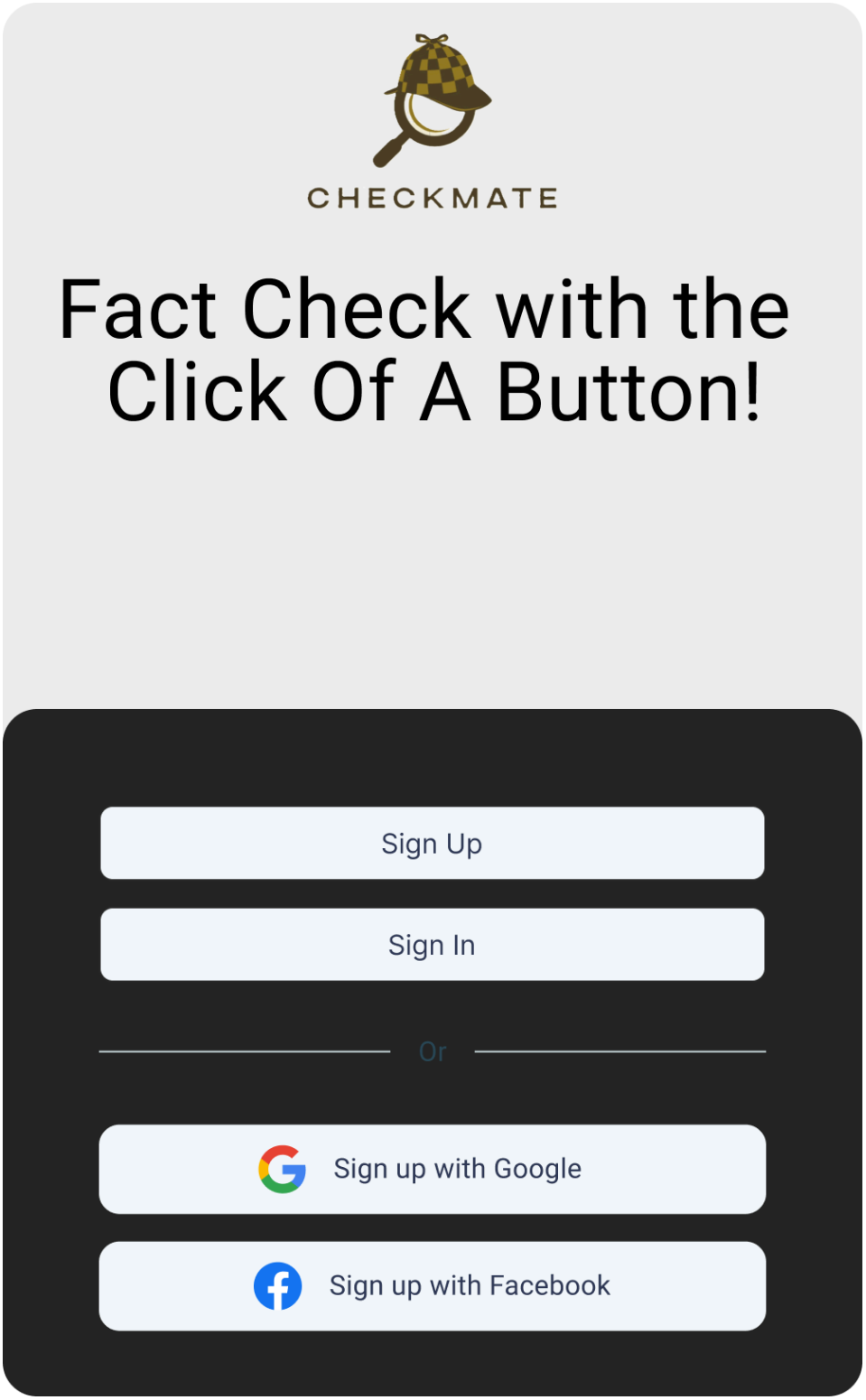
3.5.4. Dynamic Models

3.5.4.1. Sequence Model

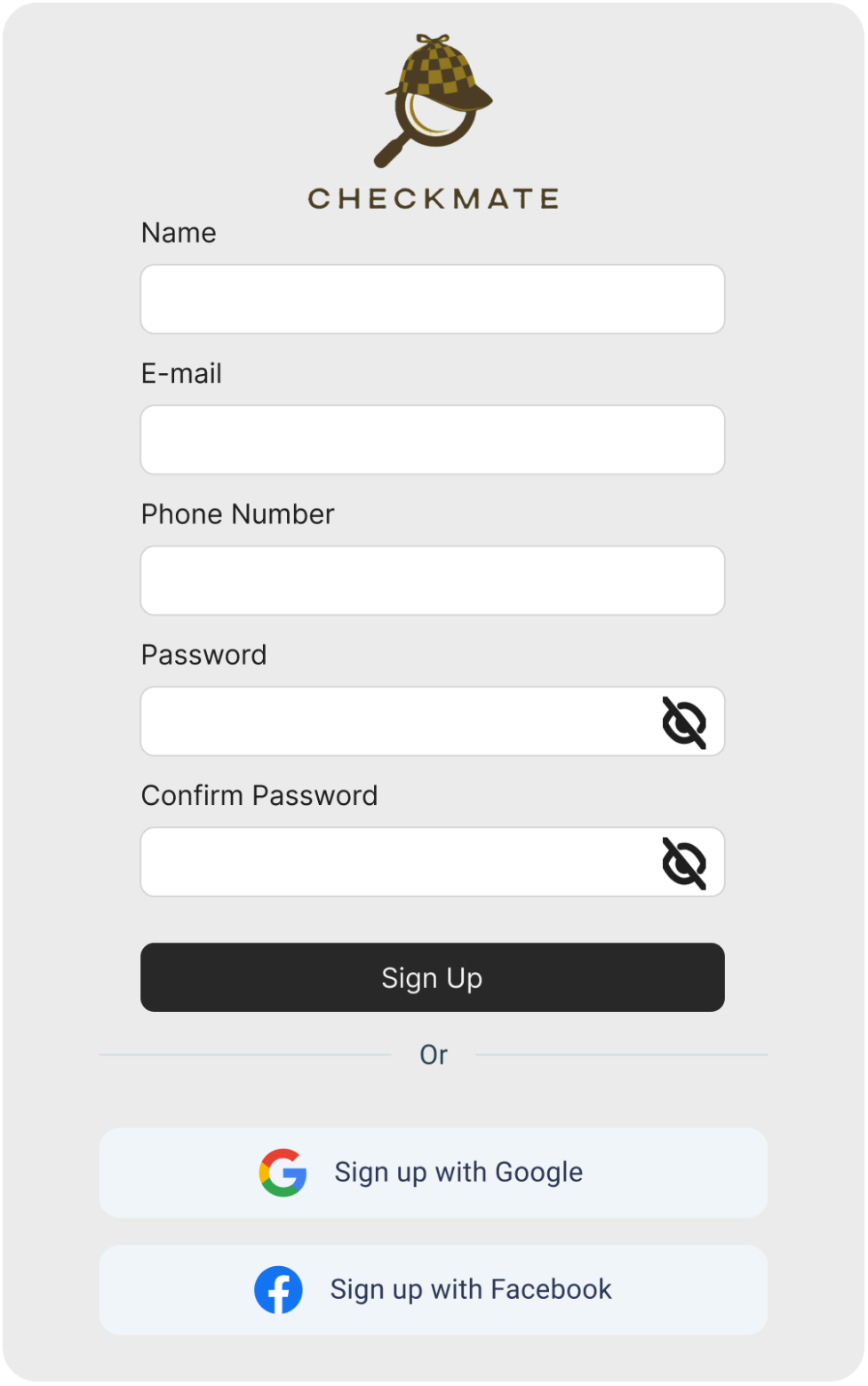


3.5.5. User Interface - Navigational Paths and Screen Mock-Ups

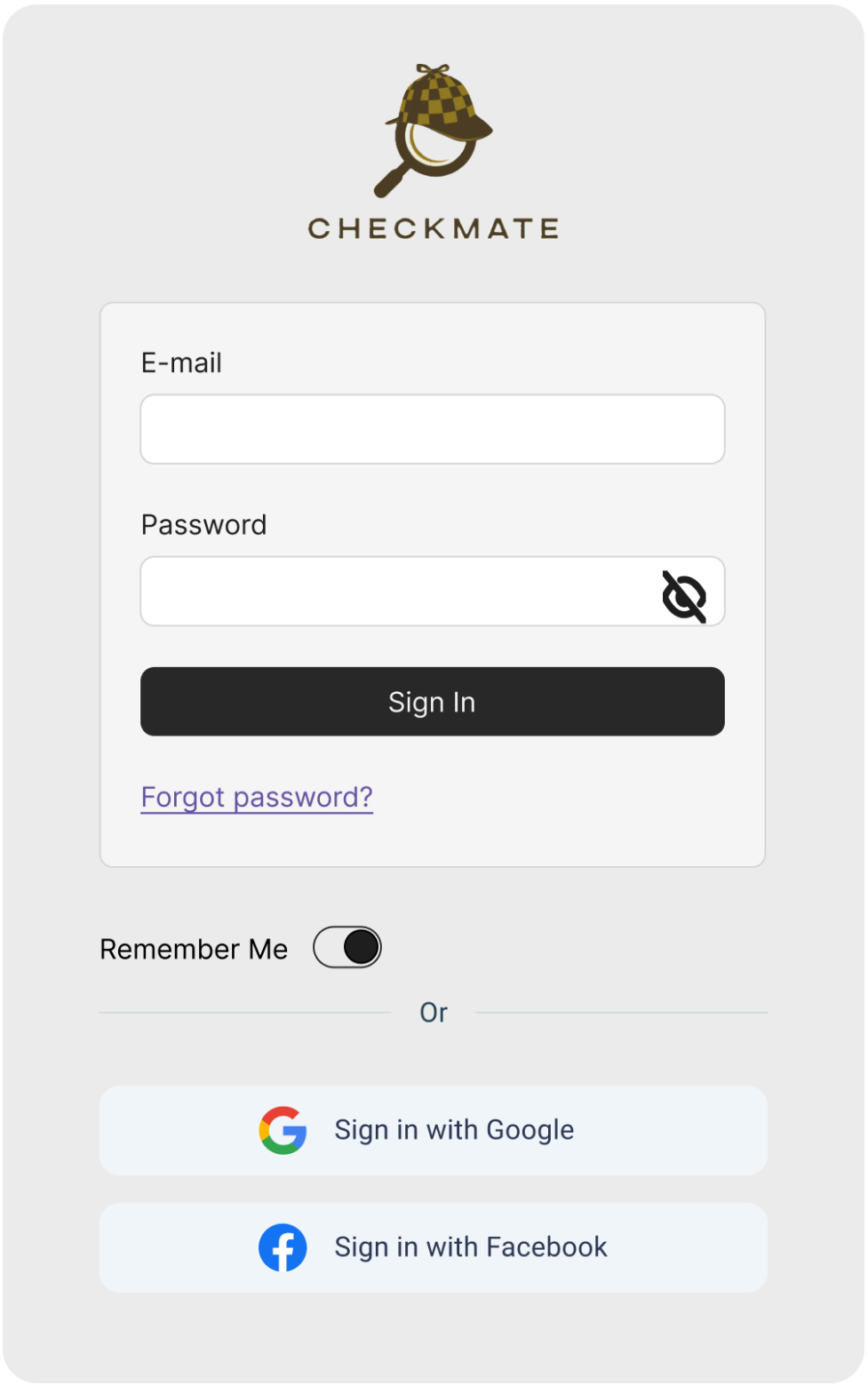
Main Menu of the extension when the user is not signed in



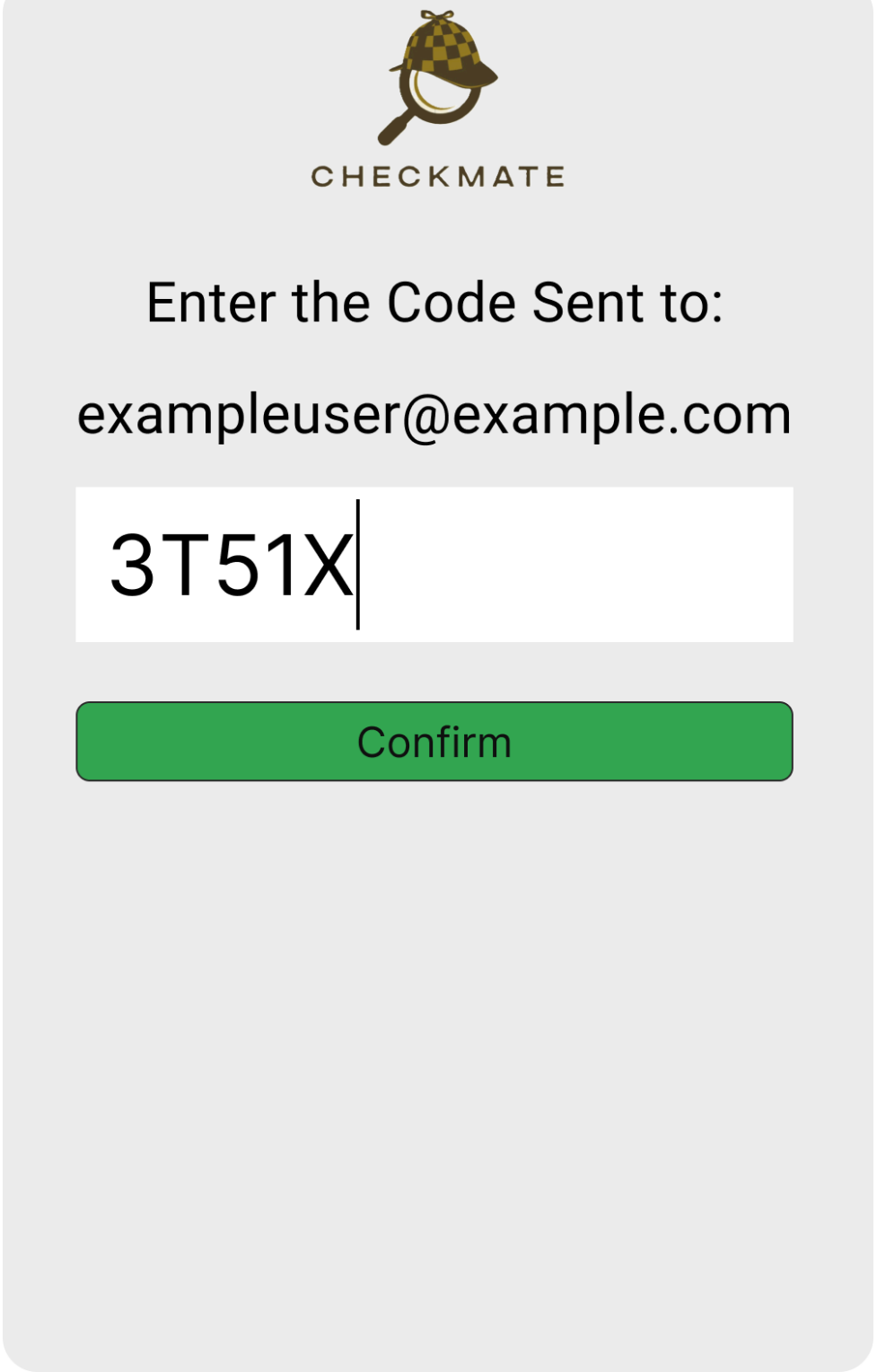
Sign Up Page


CHECKMATE
Name
E-mail
Phone Number
Password
Confirm Password
Sign Up
Or
Sign up with Google
Sign up with Facebook

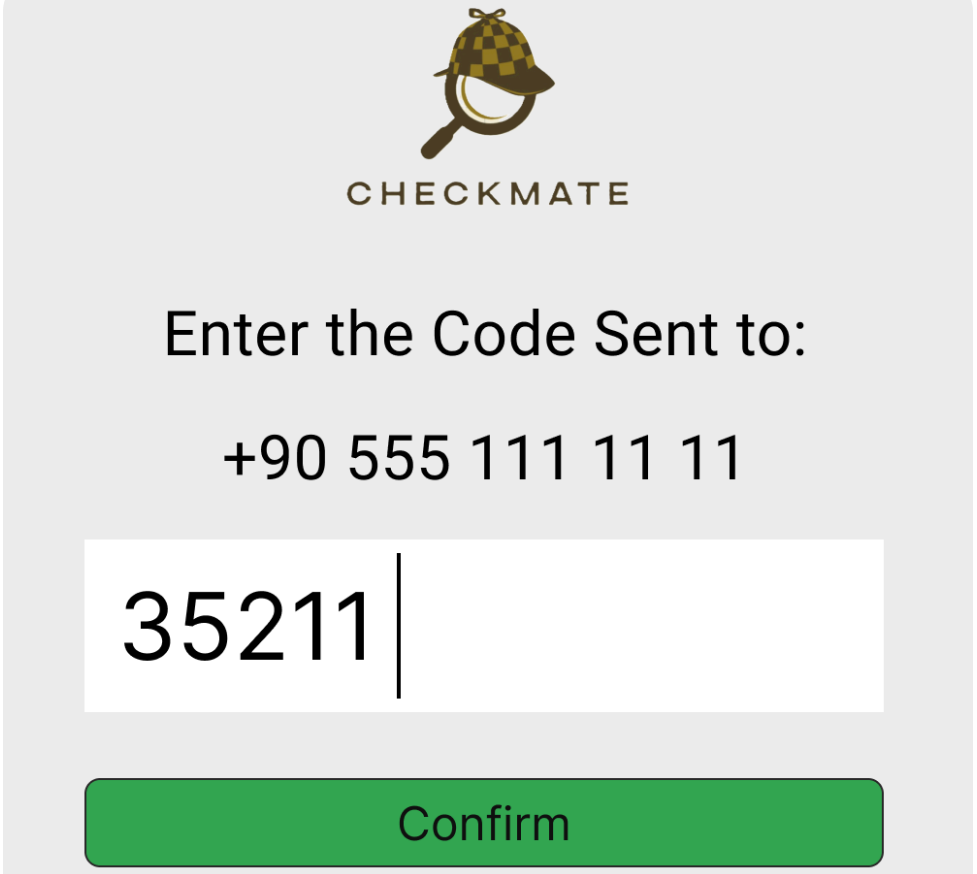
Sign In Page


CHECKMATE
E-mail
Password
Sign In
Forgot password?
Remember Me
Or
Sign in with Google
Sign in with Facebook

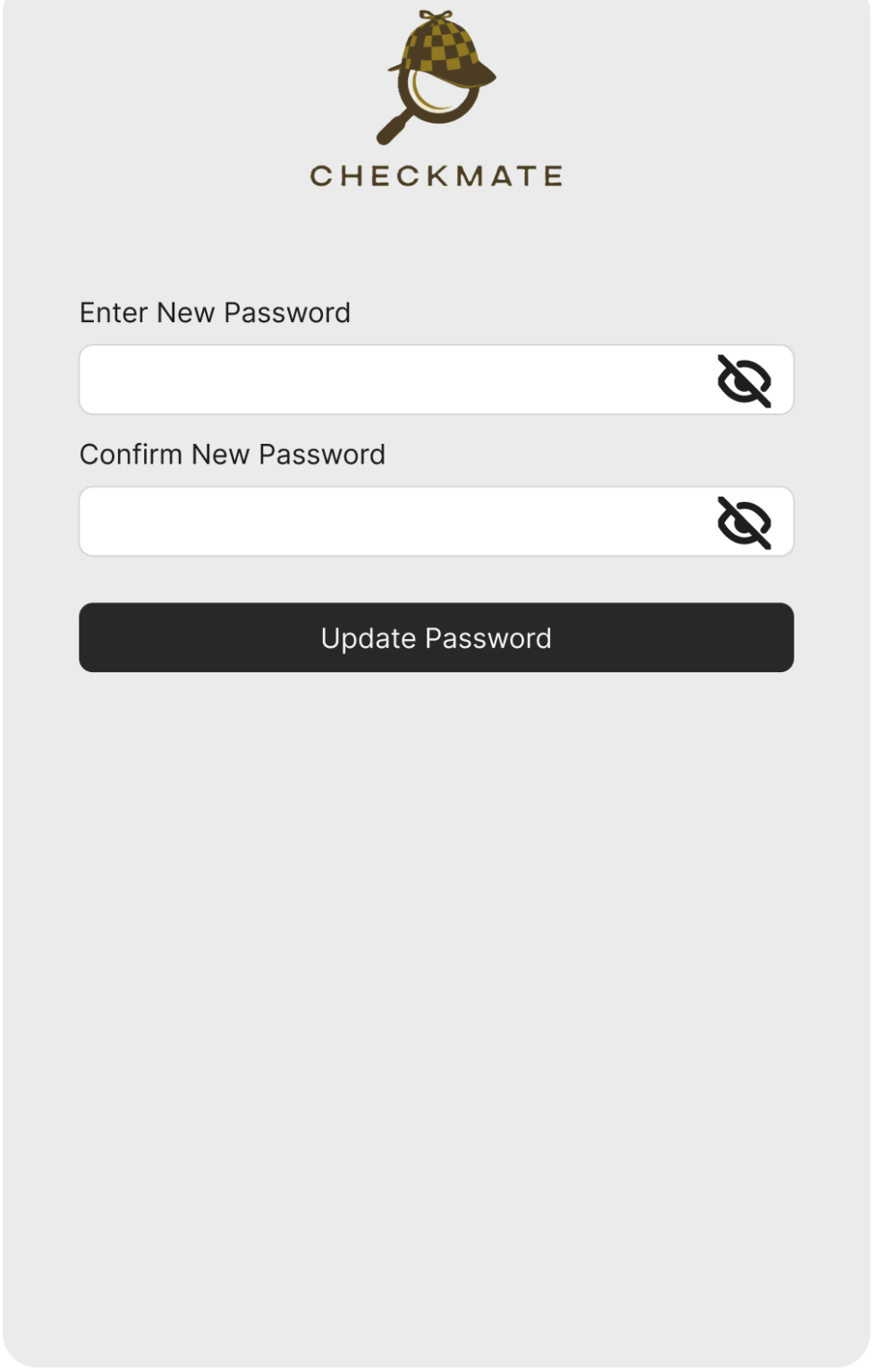
Email Verification Page



SMS Verification Page



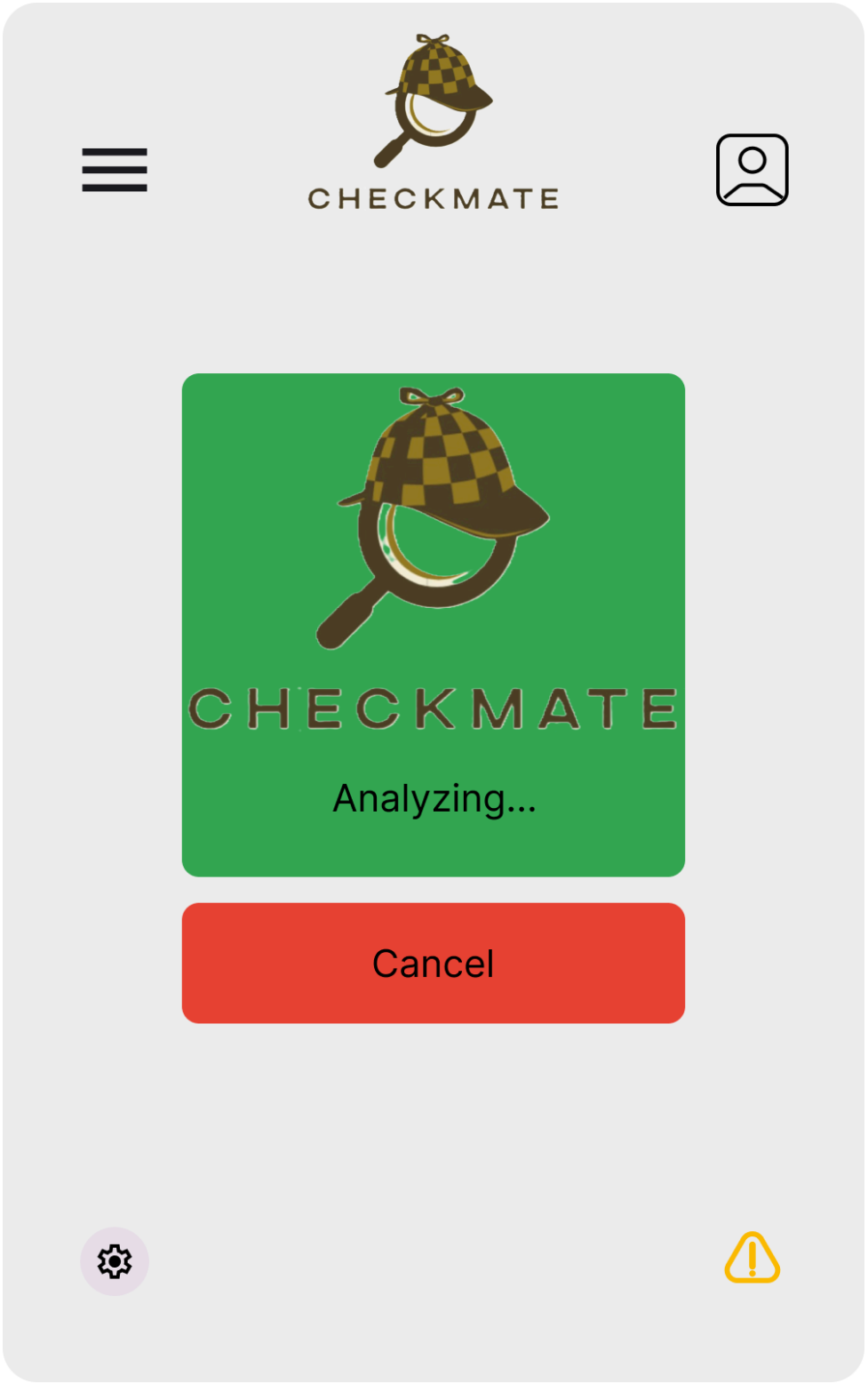
Password Reset Page



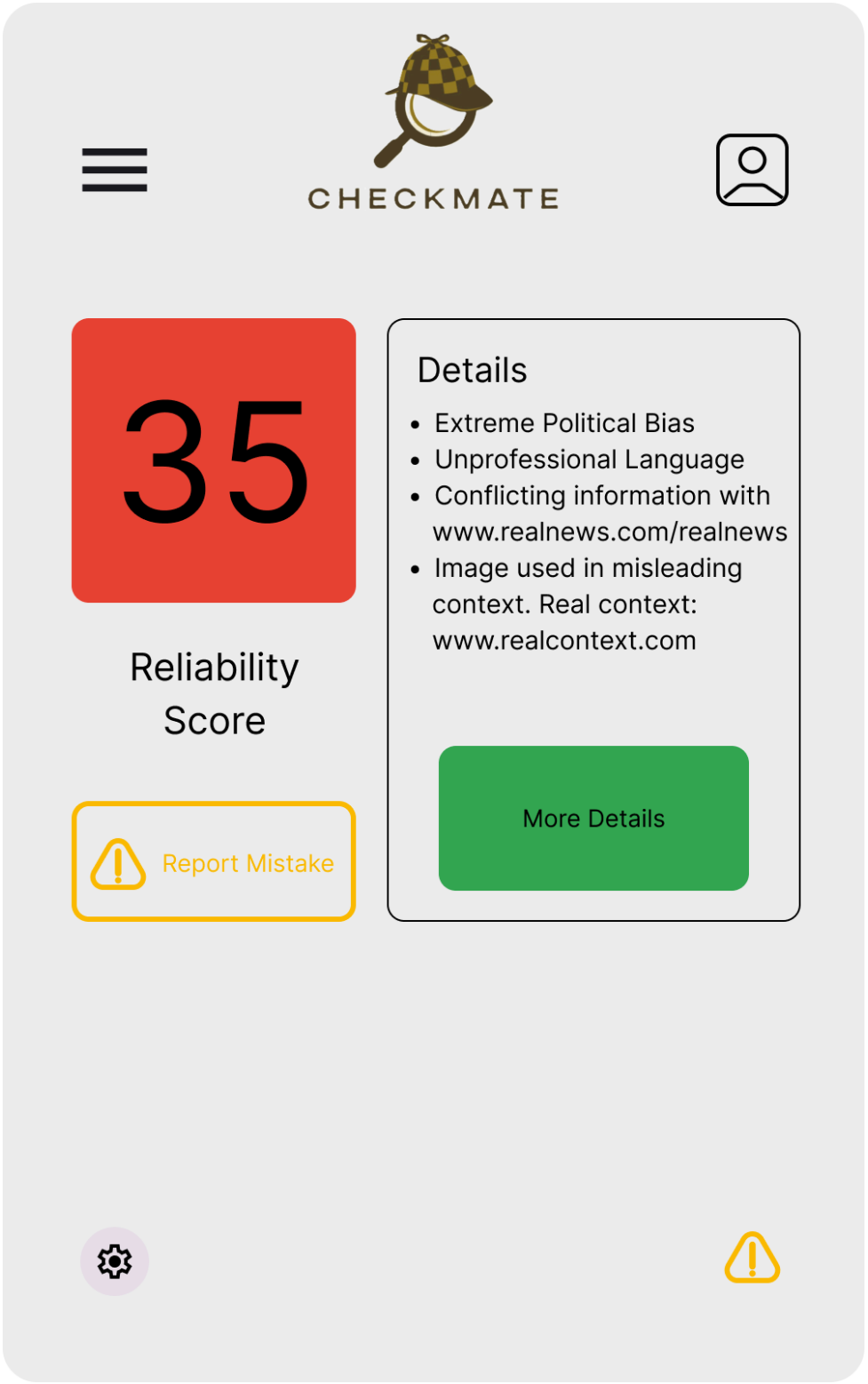
Main Menu


CHECKMATE
Analyze Current Page
news.com
Enter URL or Text to Analyze
www.news.com/news

Analyzing Page

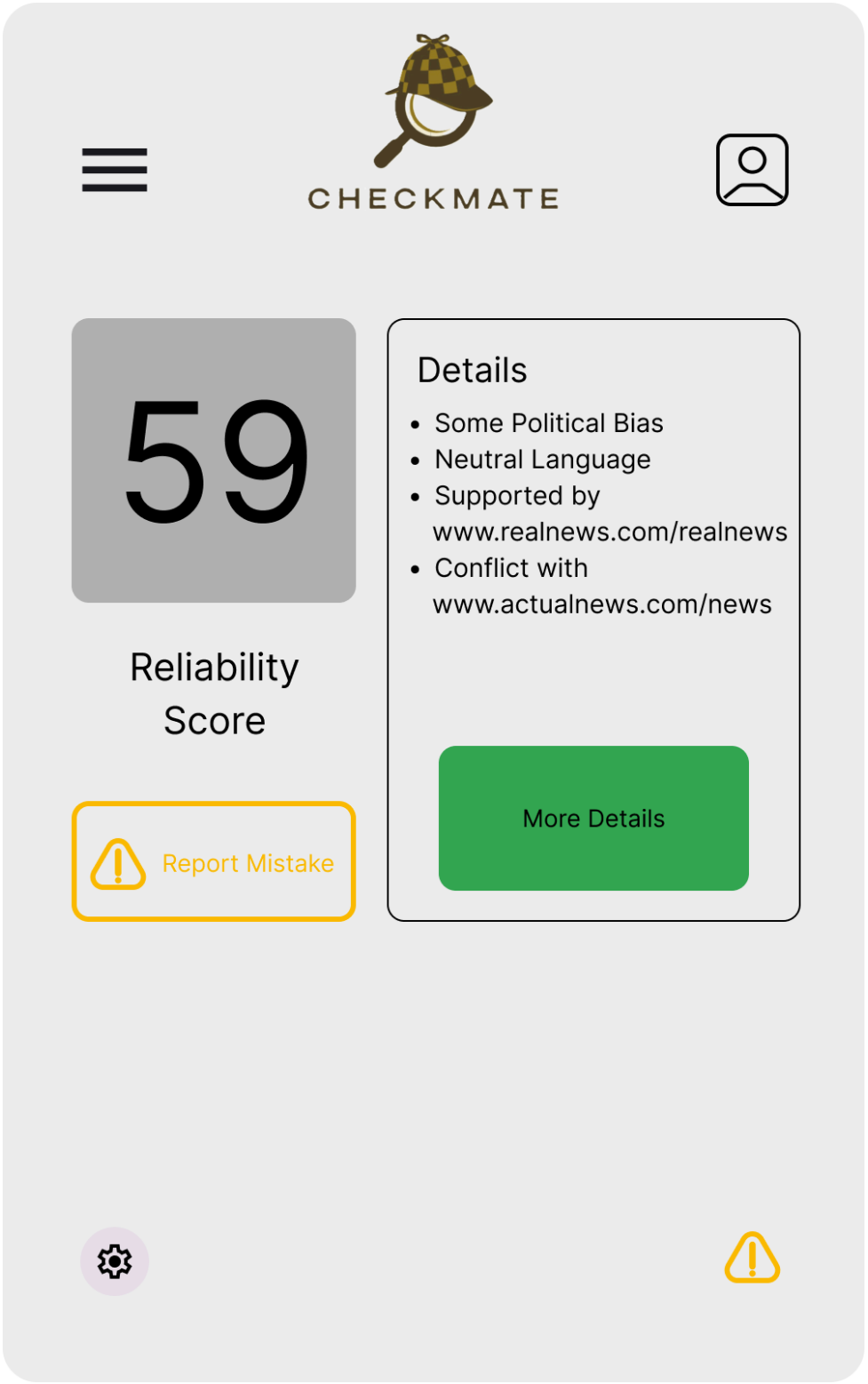


Analysis Result Page - Negative



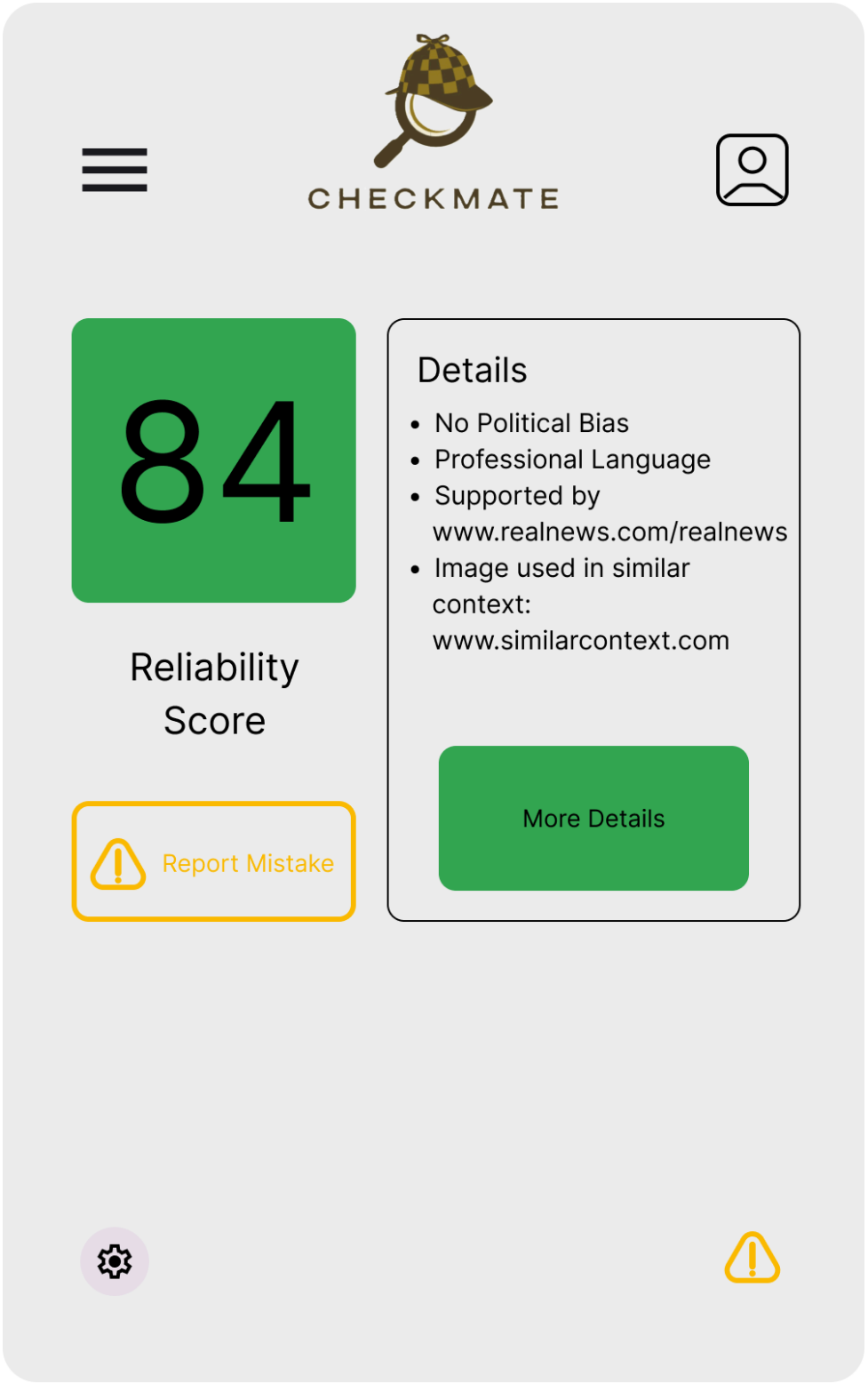
Analysis Details - Negative

Analysis Result Page - Neutral



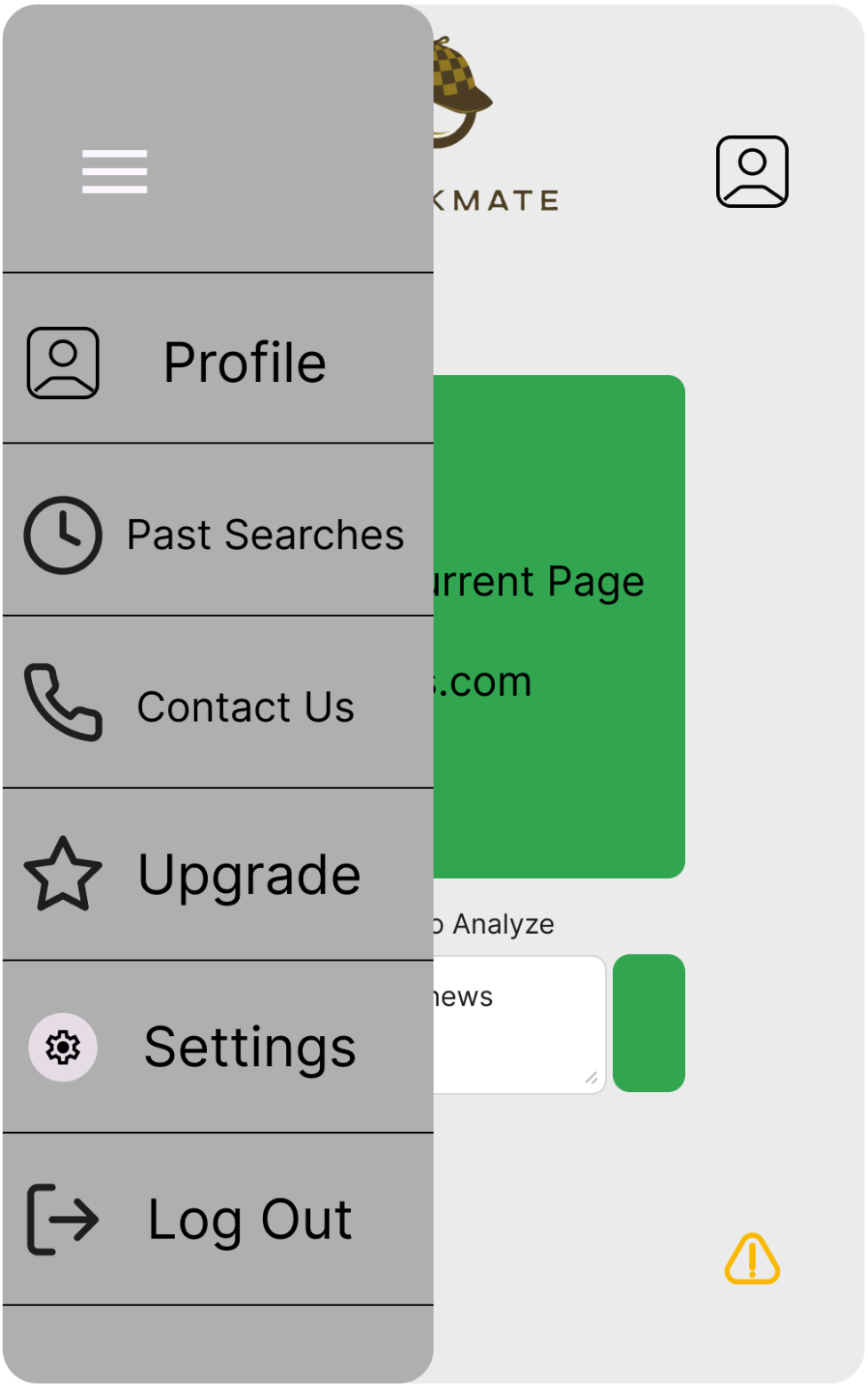
Analysis Details - Neutral

Analysis Result Page - Positive

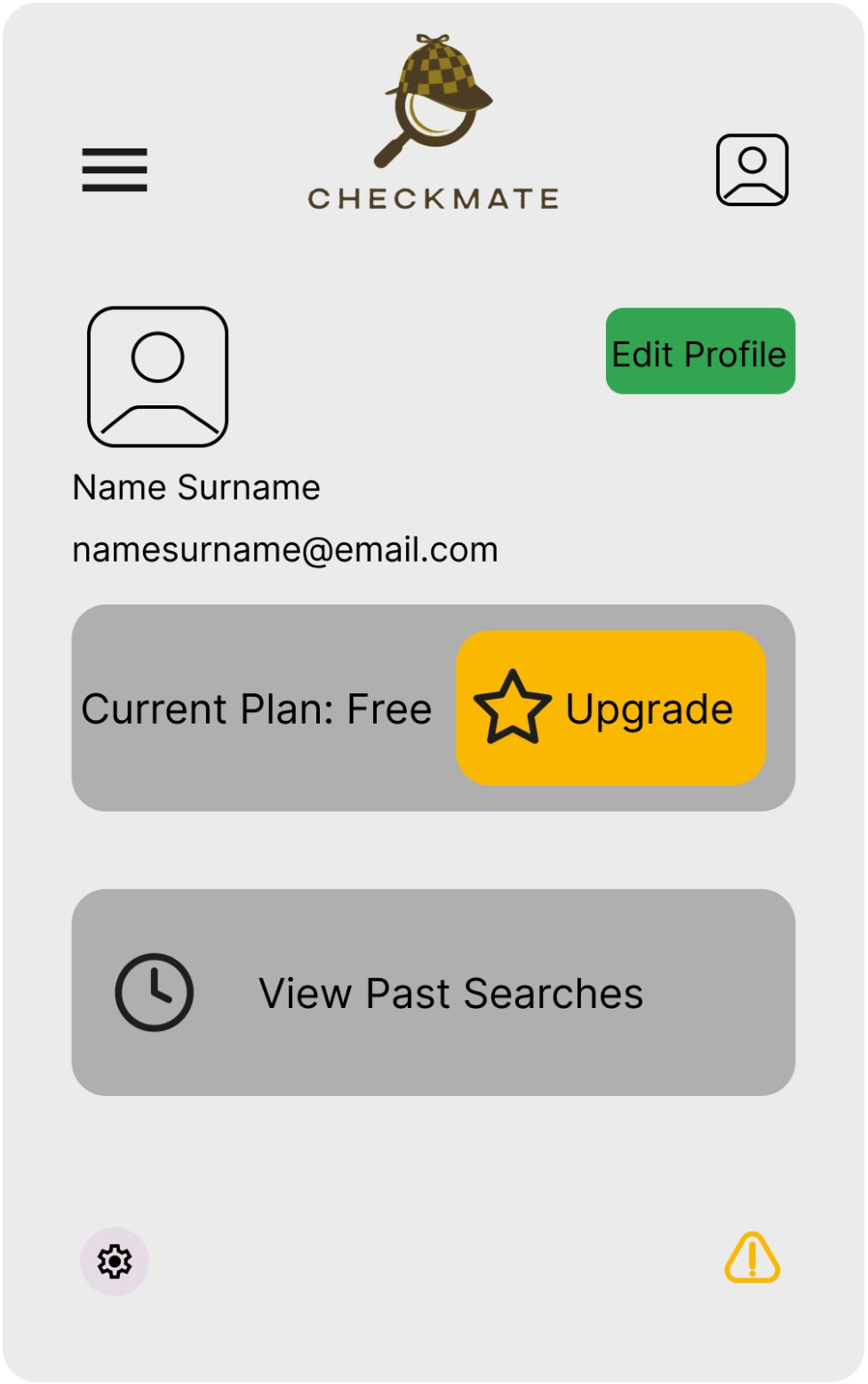


Analysis Details - Positive

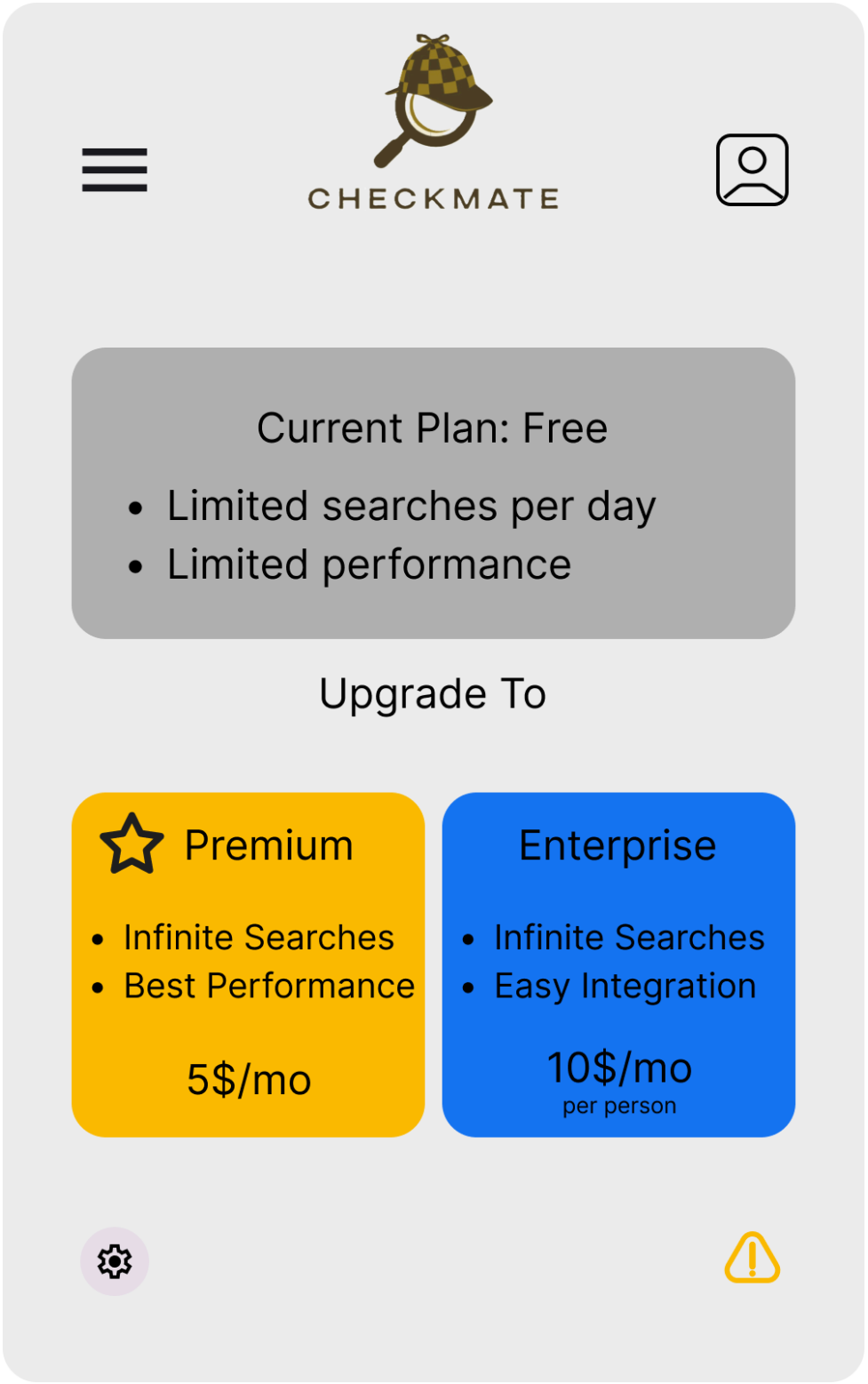
Example Sidebar



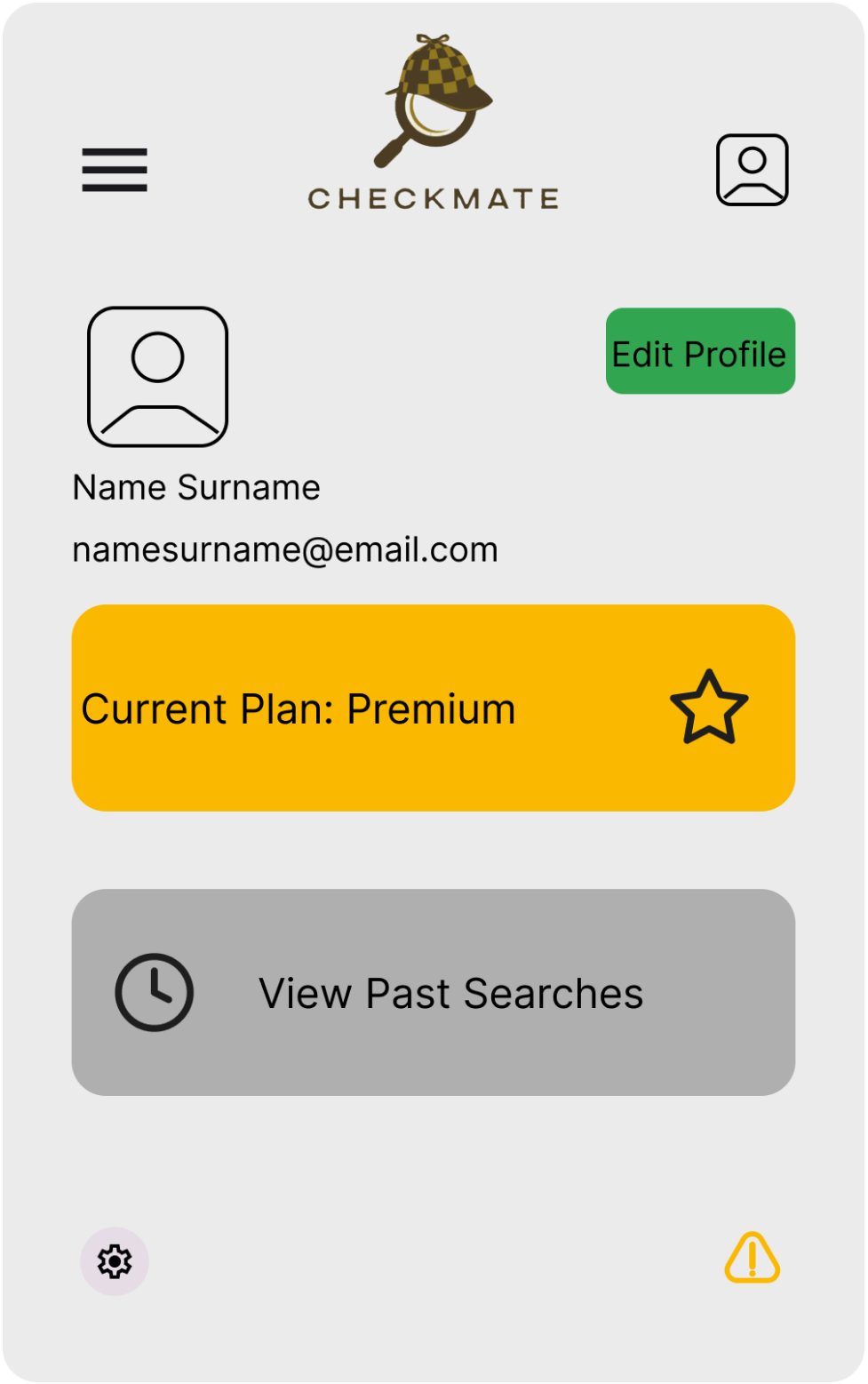
Profile Page



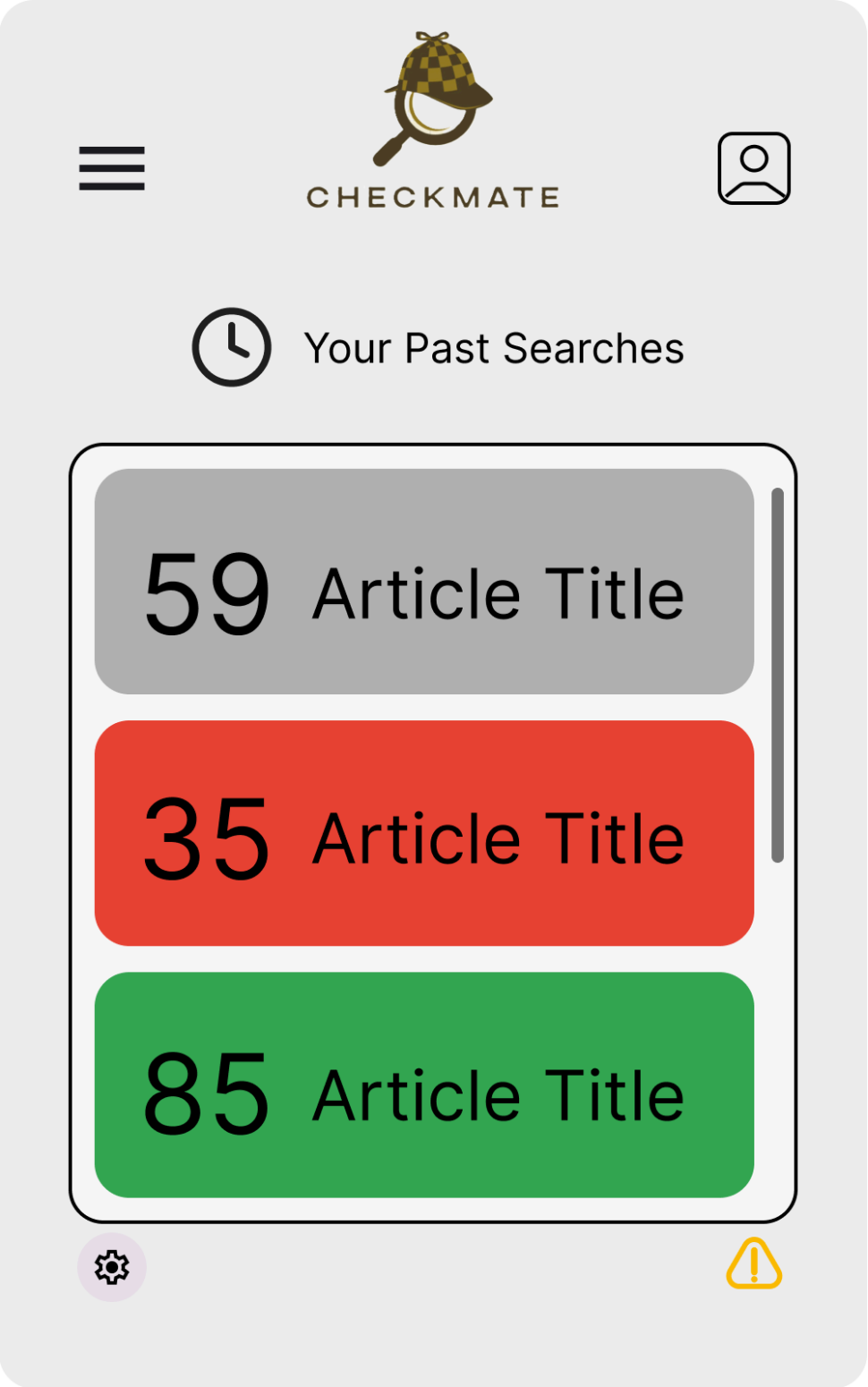
Upgrade Page



Profile Page with Premium



Past Searches



4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

The design and development of Checkmate were influenced by several key factors, including social and cultural considerations, data privacy, public health, economic stability, and global impacts. These factors shaped the system's analysis models, user interface, and overall functionality, ensuring that the application meets diverse user needs while addressing critical societal challenges.

4.1.1. Social and Cultural Factors

Checkmate leverages NLP models, including the Political Spectrum Analyzer and Language Analyzer, to deliver culturally and socially sensitive insights. These models were designed to:

- Determine whether a given article is left-leaning or right-leaning while minimizing inherent biases by training on diverse datasets that represent a wide spectrum of political viewpoints.
- Conduct language analysis to identify the tone of the article (biased - unbiased/objective - subjective), helping users better understand the underlying emotional and ideological biases.

The application's user interface further incorporates social and cultural considerations:

- User-friendly design ensures accessibility for individuals with varying levels of technological proficiency.

4.1.2. Data Privacy

Data privacy is a cornerstone of Checkmate, reflecting a commitment to safeguarding user information while complying with international standards such as the General Data Protection Regulation (GDPR). The application's privacy-centric design incorporates the following measures:

- Articles submitted by users are analyzed solely for political leaning and sentiment, with only the required data processed for these purposes.
- Analysis results are returned without storing sensitive article content, ensuring minimal data retention and reducing risks of misuse.
- Adherence to the principle of data minimization ensures that only essential information is collected and processed.
- Users are provided with clear, transparent communication about data collection practices and must provide explicit consent before any analysis occurs.

By integrating these privacy measures into its design, Checkmate ensures compliance with legal standards, reinforces user trust, and aligns with best practices for ethical data handling.

4.1.3. Public Health

Fake news has significant implications for public health, particularly during crises such as pandemics or natural disasters. Misinformation about vaccines, treatments, or public health policies can lead to panic,

mistrust in authorities, and harmful behaviors. Checkmate mitigates these risks by:

- Analyzing the credibility of health-related articles, helping users discern reliable information.
- Detecting emotional manipulation through sentiment analysis, flagging articles that may intentionally spread fear or misinformation.
- Supporting informed decision-making, ultimately improving public understanding and health outcomes.

4.1.4. Economic Factors

Fake news has profound economic impacts, including market manipulation, disruptions to industries, and financial harm to individuals. For instance, misinformation about companies or stock markets can mislead investors and create instability. Checkmate addresses these challenges by:

- Identifying misleading financial news through bias and sentiment analysis, enabling users to assess the reliability of economic information.
- Supporting businesses and policymakers in making informed decisions by providing tools to verify the credibility of financial content.

4.1.5. Global Factors

Misinformation often transcends national boundaries, influencing international relations and public opinion on global issues such as climate change, immigration, or conflicts. Checkmate accounts for

these challenges by supporting efforts to combat misinformation campaigns that could harm diplomatic relations or international initiatives.

- By addressing the global nature of misinformation, Checkmate contributes to fostering accurate discourse and collaboration across nations.

4.1.6. Impact on Analysis and Design

These factors have significantly influenced the analysis and design of Checkmate:

- The integration of NLP models for political spectrum and language analysis reflects the importance of addressing cultural diversity and bias mitigation.
- The prioritization of data privacy ensures compliance with legal standards while maintaining user trust.

Table: Aforementioned Considerations and Effect Degree

Factor	Description	Effect Degree (0-10)
Social and Cultural	Ensures culturally and socially sensitive analyses by, avoiding bias reinforcement, and supporting inclusivity through design.	10

Data Privacy	Protects user information with encryption, minimal data retention, transparency, and GDPR compliance to maintain user trust and legal alignment.	10
Public Health	Mitigates misinformation on health topics to prevent panic, mistrust, and harmful behaviors, promoting community well-being and safety.	8
Economic Factors	Identifies and mitigates misleading financial news to protect individual and market stability, fostering informed decision-making.	8
Global Factors	Combats misinformation on international issues and promotes accurate global discourse.	8

4.1.7. Constraints

4.1.7.1. Implementation Constraints

- The browser extension will be compatible with Chromium browsers. Some chromium browser examples can be found at [11].
- GitHub and Jira will be used to track the deadlines, issues, and code.
- HTML and CSS frameworks will be used for frontend development.
- Python will be used for machine learning (ML) development and backend development.

- PostgreSQL hosted by Amazon RDS server will be used for the database system.

4.1.7.2. Economic Constraints

- Our project requires several external libraries, frameworks, and models. Therefore, our group has opted to use as many open-source frameworks as possible.
- Google Vision API [12], the backend deployment on Amazon services, and Amazon Sagemaker for ML development will use a paid plan.
- If necessary, Bing's news search API [6] and News API [13] will be used for finding news and will use a paid plan.

4.1.7.3. Ethical Constraints

- All interactions and data collected from the users will be handled within data protection law General Data Protection Regulation (GDPR).
- The users must be well informed about the application's limitations and scope before registration.
- The program will be utilizing personal data and storing said data in a database hosted on Amazon RDS.
- No unnecessary user data will be collected.
- The system will clearly communicate to the user about its shortcomings.

- The system will clearly communicate reasons for reliability scores of the news articles and the reasons for credibility scores for news sources.
- The system will suggest trustworthy sources to the user about the news article that they are searching for.
- The system is designed to ensure objectivity and fairness, actively compensating for any biased outcomes to provide accurate labeling of news articles and news sources.

4.1.7.4. Language Constraints

- The system will work on news articles and new sources in English because of the lack of labeled Turkish news article datasets.

4.1.8. Standards

- The system will abide by the European Fact-Checking Standards Network (EFCSN) [14] as a benchmark in fact-checking news articles.
- The system will abide by Google Cloud Vision API Terms of Service, Bing Search API License Agreement, and other third-party API usage policies.
- The system will abide to General Data Protection Regulation (GDPR) [15]

4.2. Risks and Alternatives

4.2.1. Risks

CheckMate may face several potential risks that we must consider and plan for such as:

- Technological Risks: Potential issues with the scalability, performance, or reliability of the system, especially during high traffic periods or when incorporating new data sources and APIs.
- Data Quality Risks: Inaccuracies or biases in the reference databases, external APIs, or the team's own analysis models could lead to unreliable credibility assessments.
- User Adoption Risks: The project's success relies on user adoption and trust, which may be challenging to achieve, especially if the system's limitations are not communicated effectively.
- Regulatory and Legal Risks: Changes in data protection laws or API usage policies could impact the system's compliance and continued operation.

4.2.2. B Plan

To mitigate these risks, we will develop a "B plan" that includes the following elements:

- Technical Redundancy: Implementing load-balancing, caching, and failover mechanisms to ensure the system can handle high traffic and maintain availability.
- Data Quality Assurance: Regularly reviewing and updating the reference databases, verifying the reliability of external APIs, and continuously refining the analysis models regarding the user feedback.
- User Engagement Strategy: Developing clear and transparent communication about the system's capabilities, limitations, and intended use to build trust and encourage adoption.
- Regulatory Monitoring: Closely tracking changes in relevant laws and regulations, and preparing for potential policy updates that may require system modifications.

4.3. Project Plan

4.3.1. Project Goals

The primary goals of the CheckMate project are:

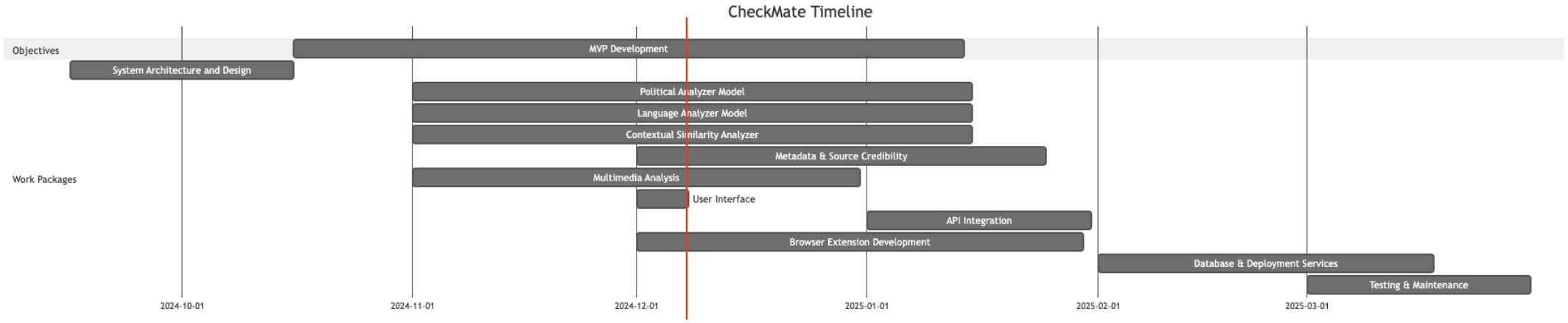
- Develop a browser extension that automatically analyzes news articles and detects potential biases and inaccuracies using AI and ML.
- Provide users with a reliable, easy-to-use tool to assess the reliability and credibility of online information.
- Contribute to the fight against the spread of misinformation and help users make more informed decisions about the content they consume.

4.3.2. Work Packages (WPs)

CheckMate will be divided into the following key work packages:

- WP1: System Architecture and Design
- WP2: Political Analyzer Model (NLP)
- WP3: Language Analyzer Model (NLP)
- WP4: Contextual Similarity Analyzer Model (NLP)
- WP5: Metadata and Source Credibility Assessment
- WP6: Multimedia Analysis and Verification
- WP7: User Interface
- WP8: API Integration
- WP9: Browser Extension
- WP10: Database and Deployment Services
- WP11: Testing, Deployment, and Maintenance

4.3.3. Gantt Chart for Project Schedule



4.3.4. Objectives and milestones for reaching the goals.

We have identified the following key objectives and milestones:

- Objective 1: Develop an MVP of the CheckMate browser extension by the end of December 2024.
    - Milestone 1.1: Complete the system architecture and design by the end of December 2024.
    - Milestone 1.2: Implement the core language analysis and bias detection algorithms by the end of December 2024
    - Milestone 1.3: Integrate the system with some of the external APIs by the end of December 2024.
- Objective 2: Integrate the system with all of external APIs and databases by the end of February 2025.
    - Milestone 2.1: Establish connections with news sources, fact-checking organizations, and multimedia analysis services by the end of February 2025.
    - Milestone 2.2: Finalize the user interface and feedback mechanisms by the end of February 2025.
    - Milestone 2.3: Implement data synchronization and verification mechanisms by the end of February 2025.
    - Milestone 2.4: Conduct extensive testing and quality assurance by the end of February 2025.

- Objective 3: Launch the CheckMate browser extension to the public by the end of April 2025.

4.4. Ensuring Proper Teamwork

Since our project consists of many components that require hard work and detailed analysis, ensuring proper teamwork is a key concept in order for us to have a functional product. We realized this earlier in the project, and that is why we have been using Jira from the very start of the project. Jira is a leading agile project management tool widely used by teams for planning, tracking, releasing, and supporting high-quality software [16]. We use its Calendar feature to add our deadlines for our tasks in order to help us visualize how much time we have left for various tasks. We also use its Scrum Board feature, which is an agile project management tool that enhances work visualization, limits ongoing work, and boosts efficiency, benefiting teams in organizing and completing their tasks effectively [17]. We have columns on this board such as “To-do,” “In Progress,” and “Done,” which represent the different steps of our tasks. Each team member is responsible for updating the progress of their tasks and adding subtasks regarding the steps of the task itself on the Scrum Board.

There are also other software that we use to ensure proper teamwork other than Jira. We are using WhatsApp and Google Meet for communication, GitHub for version control, Google Drive for working on reports and documents, keeping logs, and draw.io software to work on diagrams.

Other than using online software for teamwork, to ensure proper teamwork and steady progress on our project, we meet three days a week to

collaborate effectively. During these meetings, we discuss what tasks we plan to accomplish, aligning our efforts with the project timeline and goals. We also review what has been completed since the last meeting, sharing updates, insights, and any challenges faced. This routine helps us stay organized, maintain accountability, and ensure everyone is on the same page, fostering a collaborative environment where ideas are shared and solutions are developed collectively. It is also worth mentioning that we divide the workload evenly so that no member is doing significantly larger tasks than others. We try to ensure proper teamwork by doing all the steps mentioned above.

Every team member is given a specific task to focus on in order to function effectively. When we are writing a report, we divide it equally so that each member has a contribution to each report. We first started our project by working on APIs and NLP models since they were the core components of our project. It was crucial for us to understand the applicability of the models and the usability of the APIs from the start. After we had an understanding of both, we started drawing the user interfaces.

Ayberk is mainly developing the Browser Extension with its frontend and backend. Pelin and Batu are working on the APIs by filtering out their outputs and sending necessary requests. Efe Tuna is working on the Language Analyzer model for it to analyze given text data as “biased - unbiased/objective - subjective”. İpek is working on the Political Analyzer model for it to analyze given text data as “Left - Center - Right” as well as the Contextual Similarity Analyzer model for it to give a similarity score for two given text data.

4.5. Ethics and Professional Responsibilities

The CheckMate project team is committed to upholding ethical standards and professional responsibilities. Key considerations include:

- Protecting user privacy and data security in compliance with GDPR and other relevant regulations.
- Maintaining transparency about the system's capabilities, limitations, and potential biases.
- Ensuring the credibility assessments are fair, unbiased, and not used to suppress legitimate political views.
- Addressing user concerns and feedback in a timely and respectful manner.
- Continuously improving the system's accuracy and reliability through rigorous testing and updates.

4.6. Planning for New Knowledge and Learning Strategies

The development of Checkmate involved the use of several technologies, some of which were new to our team. While most members were familiar with foundational tools such as Python, HTML and CSS, others had limited experience with advanced technologies like BERT, the Google Vision API, and specific Python libraries used for NLP and browser extension development. Recognizing this skill gap early in the project, we adopted a proactive learning strategy to ensure successful integration of these tools into the application.

For the BERT-based models used in the Political Spectrum Analyzer and Language Analyzer, team members followed tutorials on transformer based architectures and studied BERT's official documentation. By experimenting with fine-tuning BERT for our specific use case, we gained a deeper understanding of how to adapt pre-trained models for analyzing political bias and sentiment in news articles. Additionally, we collaborated to refine hyperparameters and optimize performance, ensuring the models met the required accuracy and efficiency standards.

Incorporating the Google Vision API for the image verification feature was another key challenge. Many team members were unfamiliar with its usage, including authentication processes, API calls, and interpreting response outputs. To overcome this, we explored Google's developer documentation, completed small-scale experiments, and implemented test cases to understand the API's capabilities. This iterative process allowed us to efficiently integrate the reverse image search functionality into Checkmate.

Throughout the learning process, team members with more experience in specific technologies actively mentored others, fostering a collaborative and supportive atmosphere. We balanced the time required for skill acquisition with project deadlines, ensuring that learning new technologies did not disrupt our overall timeline. Whenever challenges arose, we relied on official documentation, online tutorials, and community forums to troubleshoot issues.

By adopting this structured approach to acquiring new knowledge, including hands-on experimentation, collaborative learning, and the use of external

resources, we ensured that the entire team was equipped to contribute effectively to the development of Checkmate.

5. Glossary

API: An API, or Application Programming Interface, is a collection of rules and protocols that allow software applications to interact and share data, features, or functionality. API communication can be understood as an exchange of requests and responses between a client and a server. The client is the application sending the request, while the server processes the request and sends back a response. The API acts as the intermediary, facilitating this connection between the two. [18]

NLP: Natural Language Processing (NLP) is a branch of computer science and artificial intelligence focused on enabling computers to comprehend human language. It combines computational linguistics, which examines the mechanics of language, with statistical methods, machine learning, and deep learning models. These techniques equip computers to analyze and interpret text or speech data, understanding their overall meaning, as well as the speaker's or writer's intentions and emotions. [19]

ML: Machine learning (ML) is a branch of artificial intelligence (AI) and computer science that focuses on using data and algorithms to enable AI to imitate the way that humans learn, gradually improving its accuracy. [20]

URL: The location of a webpage or file on the Internet. [21]

6. References

[1] Meta, "How fact-checking works," Meta Transparency Center. [Online]. Available: https://transparency.meta.com/en-us/features/how-fact-checking-works/. [Accessed: 16-Nov-2024].

[2] Knight Science Journalism, "Fact-checking science journalism: How to make sure your stories are true," KSJ Handbook. [Online]. Available: https://ksjhandbook.org/fact-checking-science-journalism-how-to-make-sure-your-stories-are-true/the-fact-checking-process/. [Accessed: 16-Nov-2024].

[3] J. Waterson, "Internet overtakes TV as UK's most popular news source for first time," The Guardian, 10-Sep-2024. [Online]. Available: https://www.theguardian.com/media/article/2024/sep/10/internet-tv-uk-most-popular-news-source-first-time. [Accessed: 16-Nov-2024].

[4] GitHub, "U.S. News Domains," GitHub. [Online]. Available: https://github.com/ercexpo/us-news-domains. [Accessed: 16-Nov-2024].

[5] Google Developers, "Custom Search API," Google Developers. [Online]. Available: https://developers.google.com/custom-search/v1/overview. [Accessed: 16-Nov-2024].

[6] Microsoft, "Bing News Search API," Microsoft. [Online]. Available: https://www.microsoft.com/en-us/bing/apis/bing-news-search-api. [Accessed: 16-Nov-2024].

[7] Microsoft, "Bing Custom Search API," Microsoft. [Online]. Available: https://www.microsoft.com/en-us/bing/apis/bing-custom-search-api. [Accessed: 16-Nov-2024].

[8] Google Cloud Vision, "Label detection," Google Cloud. [Online]. Available: https://cloud.google.com/vision/docs/labels. [Accessed: 16-Nov-2024].

[9] Google Cloud Vision, "Web detection," Google Cloud. [Online]. Available: https://cloud.google.com/vision/docs/detecting-web#vision_web_detection-python. [Accessed: 16-Nov-2024].

[10] Microsoft, "Bing Image Search API," Microsoft. [Online]. Available: http://www.microsoft.com/en-us/bing/apis/bing-image-search-api. [Accessed: 16-Nov-2024].

[11] Mihir J, “Browser Comparison Finale Chromium-based Browsers”, Medium. [Online]. Available:https://medium.com/@mihirgrand/browser-comparison-finale-chromium-based-browsers-2b6063e74165 [Accessed: 16-Nov-2024].

[12] Google, "Vision AI," Google Cloud. [Online]. Available: https://cloud.google.com/vision/?hl=en. [Accessed: 16-Nov-2024].

[13] NewsAPI, NewsAPI.org. [Online]. Available: https://newsapi.org/. [Accessed: 16-Nov-2024].

[14] EFCSN, "Code of standards," *EFCSN*. [Online]. Available: https://efcsn.com/code-of-standards/. [Accessed: 16-Nov-2024].

[15] Intersoft Consulting, “General data protection regulation (GDPR),” General Data Protection Regulation (GDPR) [Online], Available: https://gdpr-info.eu/ [Accessed: 16-Nov-2024].

[16] M. Potthast, J. Kiesel, K. Reinartz, J. Bevendorff, and B. Stein, “A Stylometric Inquiry into Hyperpartisan and Fake News,” *arXiv:1702.05638 [cs]*, Feb. 2017, Available: https://arxiv.org/abs/1702.05638

[17] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, “Fake news detection on social media: A data mining perspective,” *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, Sep. 2017, doi: https://doi.org/10.1145/3137597.3137600.

[18] IBM, "What is an API?," IBM Topics. [Online]. Available: https://www.ibm.com/topics/api. [Accessed: 16-Nov-2024].

[19] GeeksforGeeks, "Natural language processing overview," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/natural-language-processing-overview/. [Accessed: 16-Nov-2024].

[20] IBM, "What is machine learning?," IBM Topics. [Online]. Available: https://www.ibm.com/topics/machine-learning. [Accessed: 16-Nov-2024].

[21] Google, "URL," Google Support. [Online]. Available: https://support.google.com/google-ads/answer/14095?hl=en. [Accessed: 16-Nov-2024].