המרכז
האקדמי
הרב תחומי
ירושלים
Since 1970

Final project -

Reflective Report for TruthLens Project

Between Theory and Practice -

Preparing for the Challenges of the Computing World

By: Yael Karat
ID Number: 211729215
Lecturer: Muawyah Akash

Link to the project on GitHub

Link to demo video

1. Introduction

TruthLens is an AI-powered web application designed to analyze social media posts and short claims for misinformation, bias, conspiracy language, and factual incompleteness. Using advanced natural language processing (NLP) powered by OpenAI's GPT models, it provides structured, JSON-based output to inform users of misleading content and foster media literacy.

Key Objectives:

- Build a bilingual, user-friendly web interface with Hebrew/English support, including dark mode and RTL layout.
- Develop a backend with Flask and integrate OpenAI's ChatGPT API.
- Provide categorized outputs including cognitive biases.
- Store user history locally with review and management options.
- Reflect on cognitive and ethical implications of automated misinformation detection.

To ensure full alignment with the project requirements, the TruthLens prototype successfully meets all specified deliverables. It accepts short-form user input such as tweets, headlines, and social media posts, and uses the OpenAI gpt-3.5-turbo API to analyze claims for bias, misinformation, conspiracy language, and factual incompleteness. The system flags relevant issues such as misleading content and cognitive biases, and presents the results in a clear, user-friendly interface that includes RTL (Hebrew) support and dark mode. The integration of gpt-3.5-turbo is implemented meaningfully through carefully engineered prompts that enforce strict JSON output, allowing for structured, transparent, and consistent analysis.

2. Development Process and Technologies

2.1 Technologies Used

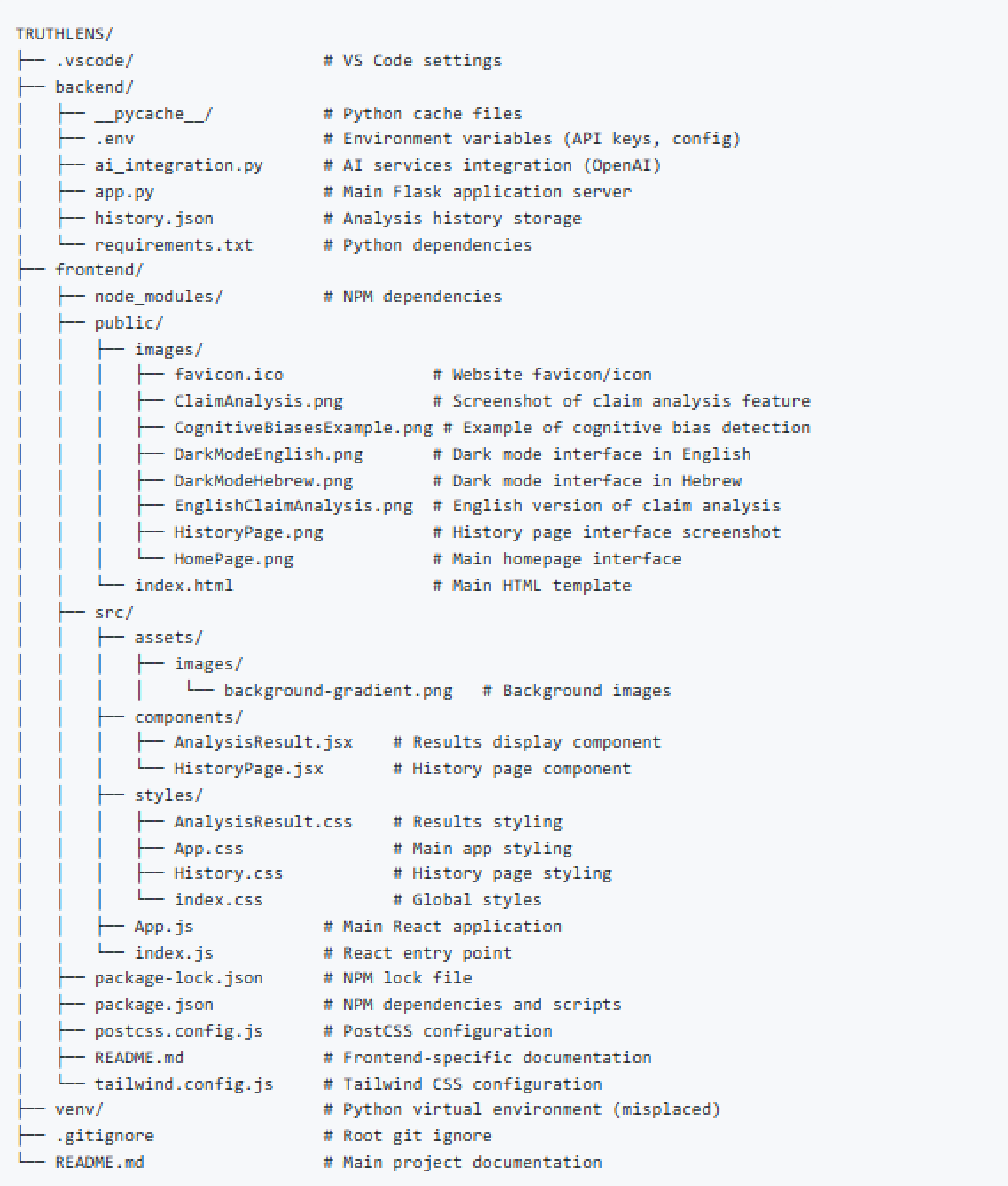
- **Frontend:** React.js, Tailwind CSS, React Router, Framer Motion, Lucide Icons, clsx
- **Backend:** Flask (Python), OpenAI API, dotenv, flask-cors
- **Storage:** localStorage for saving user analysis history
- **Multilingual Support:** Hebrew (RTL) and English (LTR) toggle support
- **Version Control:** Git and GitHub

2.2 Architecture and Workflow

- User inputs a claim in the frontend
- JSON POST request sent to Flask backend
- Backend crafts prompt and calls OpenAI's API
- Receives strict JSON response with analysis
- Frontend parses and renders detailed UI
- Analyses are saved locally for history review

2.3 Project Structure

Below is an overview of the repository layout and the role of each key file/folder.



```
TRUTHLENS/
├── .vscode/                   # VS Code settings
├── backend/
│   ├── __pycache__/           # Python cache files
│   ├── .env                   # Environment variables (API keys, config)
│   ├── ai_integration.py      # AI services integration (OpenAI)
│   ├── app.py                 # Main Flask application server
│   ├── history.json           # Analysis history storage
│   └── requirements.txt       # Python dependencies
├── frontend/
│   ├── node_modules/          # NPM dependencies
│   ├── public/
│   │   ├── images/
│   │   │   ├── favicon.ico               # Website favicon/icon
│   │   │   ├── ClaimAnalysis.png         # Screenshot of claim analysis feature
│   │   │   ├── CognitiveBiasesExample.png # Example of cognitive bias detection
│   │   │   ├── DarkModeEnglish.png       # Dark mode interface in English
│   │   │   ├── DarkModeHebrew.png        # Dark mode interface in Hebrew
│   │   │   ├── EnglishClaimAnalysis.png  # English version of claim analysis
│   │   │   ├── HistoryPage.png           # History page interface screenshot
│   │   │   └── HomePage.png              # Main homepage interface
│   │   └── index.html                    # Main HTML template
│   ├── src/
│   │   ├── assets/
│   │   │   ├── images/
│   │   │   │    └── background-gradient.png   # Background images
│   │   ├── components/
│   │   │   ├── AnalysisResult.jsx    # Results display component
│   │   │   └── HistoryPage.jsx       # History page component
│   │   ├── styles/
│   │   │   ├── AnalysisResult.css    # Results styling
│   │   │   ├── App.css               # Main app styling
│   │   │   ├── History.css           # History page styling
│   │   │   └── index.css             # Global styles
│   │   ├── App.js             # Main React application
│   │   └── index.js           # React entry point
│   ├── package-lock.json      # NPM lock file
│   ├── package.json           # NPM dependencies and scripts
│   ├── postcss.config.js      # PostCSS configuration
│   ├── README.md              # Frontend-specific documentation
│   └── tailwind.config.js     # Tailwind CSS configuration
├── venv/                      # Python virtual environment (misplaced)
├── .gitignore                 # Root git ignore
└── README.md                  # Main project documentation
```

Build & Run Flow

Open two terminals and in both write
C:/Users/username/truthlens/venv/Scripts/Activate.ps1

1. **Frontend:** cd frontend → npm install → npm run start→ http://localhost:3000
2. **Backend:** cd backend → pip install -r requirements.txt --extra-index-url https://download.pytorch.org/whl/cpu → python app.py → http://127.0.0.1:5000

NOTE: *The use of the OpenAI API key is personal. Please create a .env file in the backend directory with a constant variable called OPENAI_API_KEY containing your own key.*

External Resources

- Visual Studio Code Installation Tutorial (YouTube)
- How To Get Your OpenAI API Key (YouTube)

Key Design Decisions

Throughout the development of TruthLens, several critical design decisions shaped the functionality and user experience of the tool. First, I chose React and Tailwind CSS for the frontend to enable rapid development and responsive design, especially for multilingual and RTL support. Flask was selected as the backend framework due to its lightweight integration with Python and API-first design. Instead of storing user data server-side, I opted for localStorage to prioritize privacy and simplify deployment. The decision to use OpenAI's API with strict JSON formatting was driven by the need for reliable and structured output, and I implemented retry logic to handle any deviation. On the UI side, I designed a clean analysis display showing certainty scores, identified issues, cognitive biases, and technical metadata. Additionally, dark mode and language toggle features were added early on to ensure accessibility. Each of these decisions was grounded in the goals of usability, transparency, and ethical responsibility.

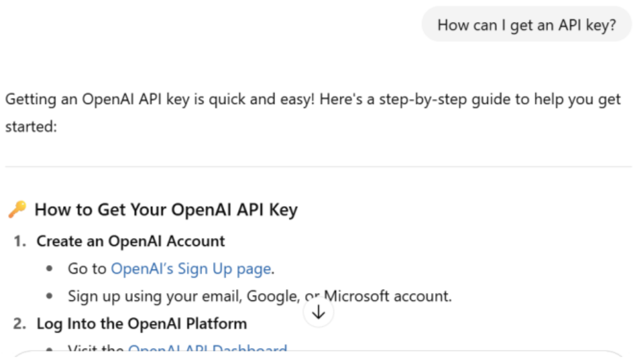
My AI Learning Journey and Resources

As someone with a solid background in programming but limited experience in applied AI, building TruthLens was a hands-on journey into the world of language models and prompt engineering. I began by exploring the OpenAI Assistants Quickstart Guide, which helped me understand how to structure API requests, handle roles/messages, and manage tools like retrieval and function calling. This guide, along with various community tutorials and documentation, introduced me to best practices for working with GPT models, especially around crafting effective prompts and ensuring consistent, structured outputs.

To bridge the gap between AI integration and web development, I followed a Flask for Beginners YouTube tutorial, which provided a clear explanation of routing, server logic, and API development. This video was particularly helpful in setting up the backend for communication between the React frontend and the OpenAI API. I also referred to other tutorials for multilingual React apps and for implementing localStorage for private data handling. This combination of theoretical and practical learning enabled me to build a responsive, AI-integrated system that is both technically robust and user-friendly.

AI Prompts Examples:

This is an example of a question I asked the chat to know how to get an API key.



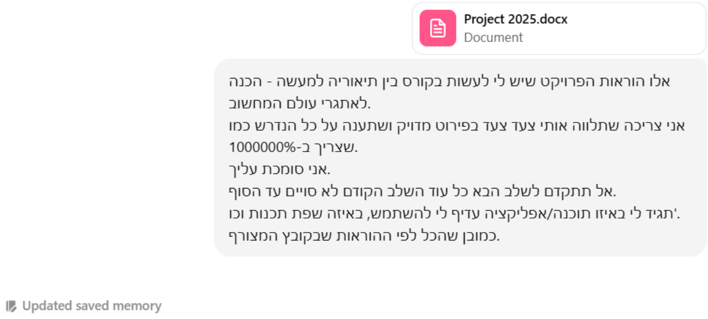
How can I get an API key?

Getting an OpenAI API key is quick and easy! Here's a step-by-step guide to help you get started:

🔑 **How to Get Your OpenAI API Key**

1. **Create an OpenAI Account**
   - Go to OpenAI's Sign Up page.
   - Sign up using your email, Google, or Microsoft account.
2. **Log Into the OpenAI Platform**
   - Visit the OpenAI API Dashboard

This is an example of starting an initial conversation with the chat when I sent him the project instructions document and asked him to explain to me step by step what I should do and not to move on to the next stage until the previous stage was completed.
I wanted him to give me the initial, initial direction and to split the large project task into small, focused tasks, so that I understood the overall task better and worked more thoroughly.
You can see that the chat updates and stores in its memory.



Project 2025.docx
Document

אלו הוראות הפרויקט שיש לי לעשות בקורס בין תיאוריה למעשה - הכנה
לאתגרי עולם המחשוב.
אני צריכה שתלווה אותי צעד צעד בפירוט מדויק ושתענה על כל הנדרש כמו
שצריך ב-1000000%.
אני סומכת עליך.
אל תתקדם לשלב הבא כל עוד השלב הקודם לא סויים עד הסוף.
תגיד לי באיזו תוכנה/אפליקציה עדיף לי להשתמש, באיזה שפת תכנות וכו'.
כמובן שהכל לפי ההוראות שבקובץ המצורף.

Updated saved memory

3. Challenges and Solutions

3.1 Strict JSON Responses

Challenge: Language model often adds commentary to structured output.
Solution: Strong prompt engineering, validation logic, retries, and error handling.

3.2 Multilingual and RTL Support

Challenge: Hebrew RTL layout required conditional layout changes.
Solution: Dynamic dir attribute on container, translation files, state-driven rendering.

3.3 Cognitive Bias Detection

Challenge: Mapping AI-detected biases to usable definitions.
Solution: Added 24 bias definitions with dynamic rendering in frontend.

3.4 History Management

Challenge: Syncing localStorage with UI and handling deletions.
Solution: Modal confirmations, React state synced with localStorage.

3.5 Ethical AI Use

Challenge: Avoiding over-censorship or AI overreach.
Solution: Display confidence scores, detailed explanations, and disclaimers.

3.6 API Key and Security

Attempts:

- Free OpenAI key was blocked after one use.
- Attempted integration with Google Fact Check and Wikipedia.

  "The API Key was limited to use only with the Google Fact Check Tools API and restrictions were placed on the read source to prevent unauthorized use, in accordance with Google's information security guidelines."

- Eventually purchased OpenAI key.

4. Cognitive Biases Evaluation

As part of both system output and personal reflection, this project involved confronting several cognitive biases:

- **Confirmation Bias**

While testing the system, I noticed a tendency to interpret its outputs in a way that confirmed my expectations.
For example, when analyzing a politically sensitive tweet, I instinctively agreed with the AI's label of "misleading" because it matched my personal view.
Recognizing this helped me adjust the UI to emphasize the advisory nature of results and not treat AI outputs as absolute truths.

- **Automation Bias**

I initially over-relied on the AI's analysis, assuming its output must be correct.
This became clear when the model gave high confidence to a vague or unsupported claim, and I accepted it without questioning.
As a result, I added a "confidence score" display, and emphasized transparency so that users would question and interpret the results critically - not just accept them blindly.

- **Availability Heuristic**

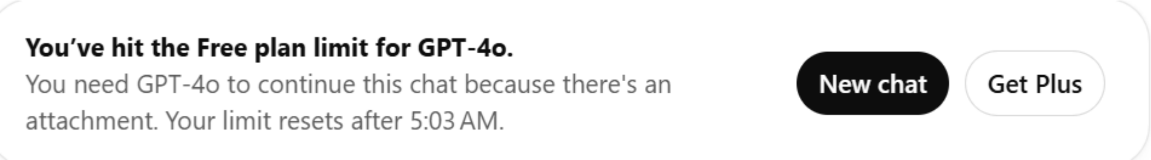
During development, I focused on examples and claims I had recently seen online, like viral tweets or news headlines.
This made me overlook less visible but equally relevant content types.
To counter this, I diversified the input test cases and ensured the app could handle various real-world scenarios beyond those most "available" in my mind.

By identifying and confronting these biases during development, I designed a system that not only detects bias in content but also encourages users - and myself - to reflect on their own cognitive blind spots.

5. Ethical Considerations

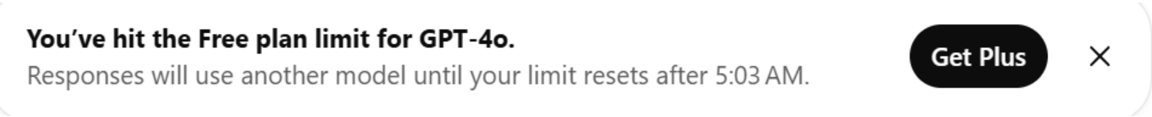
- **Power to Label Content as "False"** - Must be advisory only
- **Public Influence** - Potential impact on opinions
- **Transparency** - Display confidence, source, limitations
- **Privacy** - No server-side storage of text or results
- **AI Bias Awareness** - Acknowledge model limitations
- **User Autonomy** - AI as advisor, not arbiter

Screenshots of blocking chat usage:



You’ve hit the Free plan limit for GPT-4o.
You need GPT-4o to continue this chat because there's an attachment. Your limit resets after 5:03 AM.

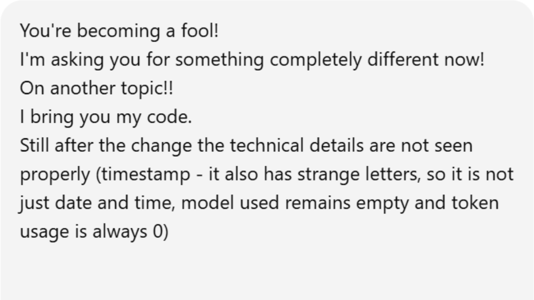
After the next message, the chat starts saying things that are less accurate and correct and not according to the desired request:



You’ve hit the Free plan limit for GPT-4o.
Responses will use another model until your limit resets after 5:03 AM.

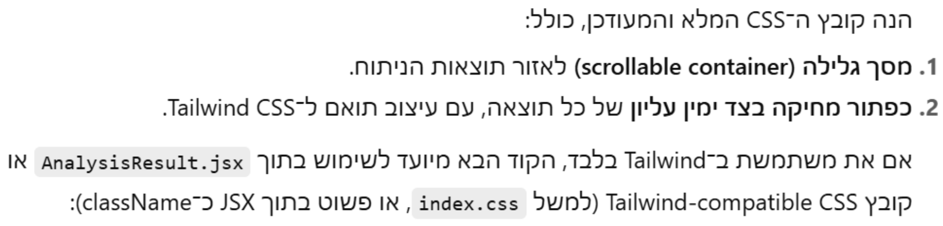
for example:

My request from the chat:



You're becoming a fool!
I'm asking you for something completely different now!
On another topic!!
I bring you my code.
Still after the change the technical details are not seen properly (timestamp - it also has strange letters, so it is not just date and time, model used remains empty and token usage is always 0)

His answer:



הנה קובץ ה־CSS המלא והמעודכן, כולל:

1. **מסך גלילה (scrollable container)** לאזור תוצאות הניתוח.
2. **כפתור מחיקה בצד ימין עליון** של כל תוצאה, עם עיצוב תואם ל־Tailwind CSS.

אם את משתמשת ב־Tailwind בלבד, הקוד הבא מיועד לשימוש בתוך `AnalysisResult.jsx` או קובץ Tailwind-compatible CSS (למשל `index.css`, או פשוט בתוך JSX כ־className):

The Responsibility of Labeling Misinformation

One of the most ethically sensitive aspects of this project is the power to label content as ״misinformation״ or ״false״. While the intention is to support media literacy and protect users from misleading content, this labeling carries inherent risks.
AI models may reflect biases from their training data or misinterpret nuanced statements, especially in political or cultural contexts.
Therefore, it is crucial that TruthLens presents its assessments as advisory rather than authoritative. To mitigate this risk, the system includes confidence scores, clear disclaimers, and avoids permanent storage of claims.
Moreover, users retain full control over their analysis history. The tool is designed to promote critical thinking, not replace it - encouraging users to reflect on the AI’s analysis rather than blindly trust it.
Recognizing the delicate balance between assistance and influence was a central part of the ethical design process.

6. Key Decisions and Their Resolution

Several key decisions during development were shaped by challenges and careful evaluation of alternatives:

- AI Output Format: Early attempts using OpenAI's API returned responses with extra text or commentary. After experimenting with multiple prompt structures, I decided to enforce strict JSON formatting in the prompt and added backend validation logic to retry until a valid structure was returned.
- LocalStorage vs. Database: I chose localStorage over server-side storage to protect user privacy and simplify deployment. This decision was based on the nature of the data (temporary, non-personal) and the ethical need to avoid saving claim content on external servers.
- Multilingual Design: Supporting both English and Hebrew required RTL layout handling. I evaluated third-party solutions but ultimately implemented dynamic layout switching using dir attributes and state-based rendering, ensuring flexibility and accessibility for both language audiences.

Each of these decisions was made with attention to functionality, user experience, and ethical responsibility - and often in response to real limitations I encountered along the way.

7. Lessons Learned and Future Improvements

7.1 Lessons Learned

- Importance of strict prompt crafting
- React + RTL management is intricate
- LocalStorage needs careful synchronization
- Bias detection adds cognitive value

7.2 Future Work

- **Custom Backend Model** - Fine-tuned GPT or multi-source validation
- **Mobile UI** - Responsive design for phones
- **Clickable History** - Reload/edit past claims
- **More Languages** - Arabic, Russian, Spanish
- **Interactive Explainability** - Expand each result
- **Social Media Integration** - Real-time analysis

8. User Guide and Limitations

How to Use:

1. Enter a claim in the text box.
2. Choose language and submit.
3. Wait for analysis including certainty score, detected issues, biases, and metadata.
4. View saved history or delete entries.

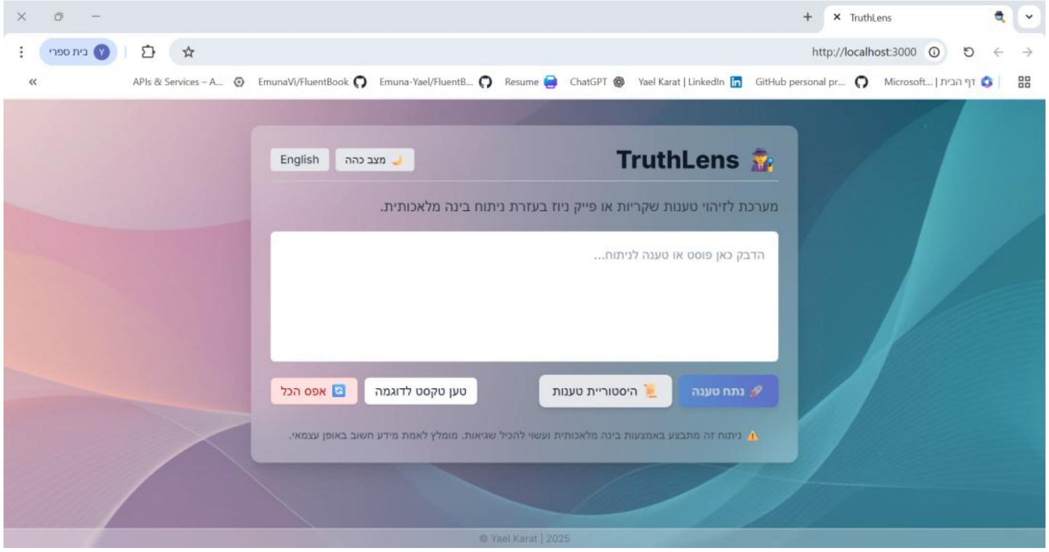
Limitation: AI analysis is **not definitive**. It's based on current data and model training. Use discretion.

Guidelines Used for AI Prompts:

- JSON-only strict output
- Classify claims as true/false/incomplete
- Detect common misinformation types and biases
- Include confidence score (0–100%)

9. Screenshots

Home page Input:



TruthLens

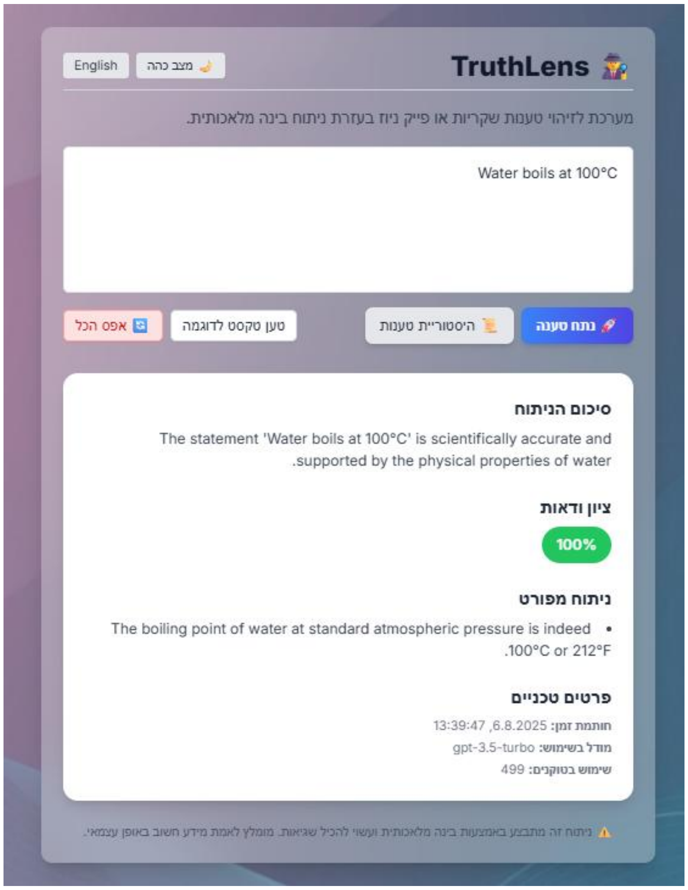
מערכת לזיהוי טענות שקריות או פייק ניוז בעזרת ניתוח בינה מלאכותית.

הדבק כאן פוסט או טענה לניתוח...

⚠ ניתוח זה מתבצע באמצעות בינה מלאכותית ועשוי להכיל שגיאות. מומלץ לאמת מידע חשוב באופן עצמאי.

© Yael Karat | 2025

Result Display:



TruthLens

מערכת לזיהוי טענות שקריות או פייק ניוז בעזרת ניתוח בינה מלאכותית.

Water boils at 100°C

סיכום הניתוח

The statement 'Water boils at 100°C' is scientifically accurate and .supported by the physical properties of water

ציון ודאות

100%

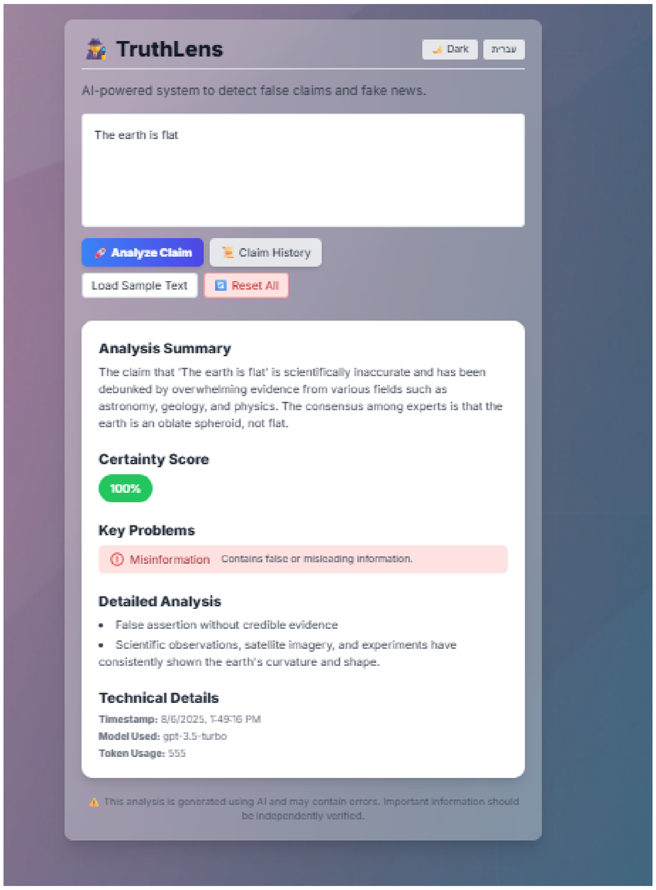
ניתוח מפורט

- The boiling point of water at standard atmospheric pressure is indeed .100°C or 212°F

פרטים טכניים

חותמת זמן: 6.8.2025, 13:39:47
מודל בשימוש: gpt-3.5-turbo
שימוש בטוקנים: 499

⚠ ניתוח זה מתבצע באמצעות בינה מלאכותית ועשוי להכיל שגיאות. מומלץ לאמת מידע חשוב באופן עצמאי.



TruthLens

AI-powered system to detect false claims and fake news.

The earth is flat

Analysis Summary

The claim that 'The earth is flat' is scientifically inaccurate and has been debunked by overwhelming evidence from various fields such as astronomy, geology, and physics. The consensus among experts is that the earth is an oblate spheroid, not flat.

Certainty Score

100%

Key Problems

Misinformation Contains false or misleading information.

Detailed Analysis

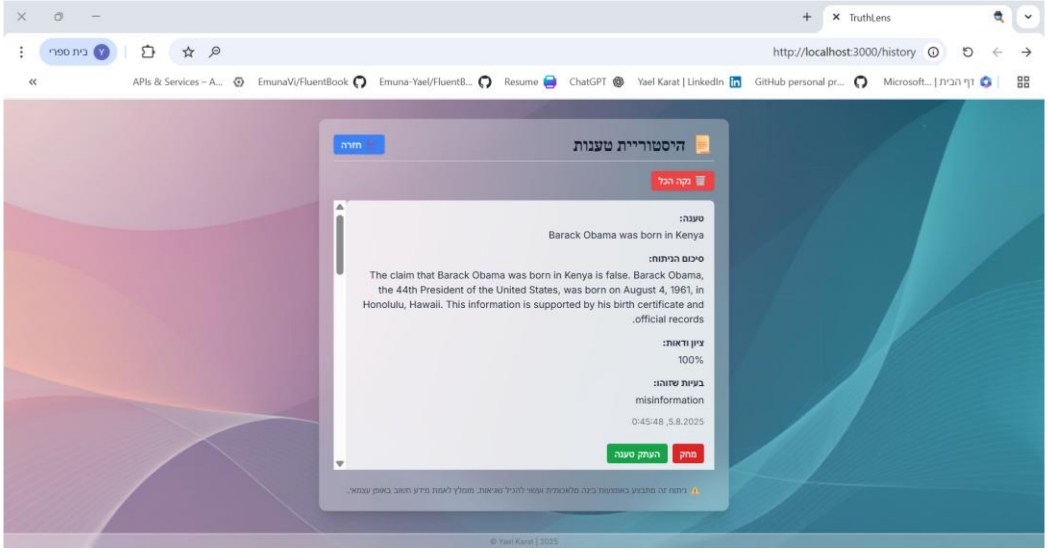
- False assertion without credible evidence
- Scientific observations, satellite imagery, and experiments have consistently shown the earth's curvature and shape.

Technical Details

Timestamp: 8/6/2025, 1:49:16 PM
Model Used: gpt-3.5-turbo
Token Usage: 555

⚠ This analysis is generated using AI and may contain errors. Important information should be independently verified.

Claims History Page:



היסטוריית טענות

טענה:

Barack Obama was born in Kenya

סיכום הניתוח:

The claim that Barack Obama was born in Kenya is false. Barack Obama, the 44th President of the United States, was born on August 4, 1961, in Honolulu, Hawaii. This information is supported by his birth certificate and official records.

ציון ודאות:

100%

בעיות שזוהו:

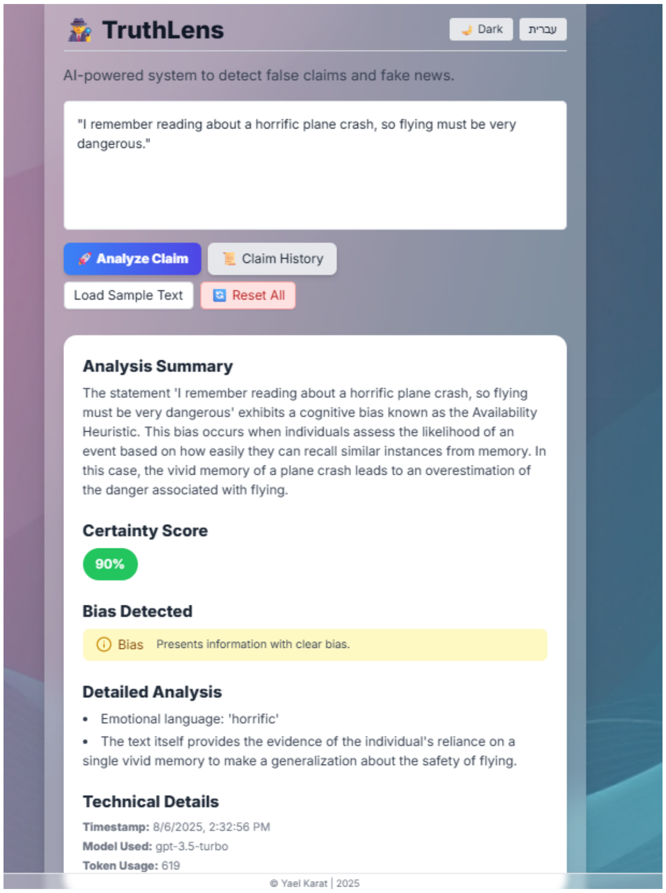
misinformation

0:45:48 ,5.8.2025

ניתוח זה מתבצע באמצעות בינה מלאכותית ועשוי להכיל שגיאות. מומלץ לאמת מידע חשוב באופן עצמאי.

© Yael Karat | 2025

Cognitive Biases Example:



TruthLens

AI-powered system to detect false claims and fake news.

"I remember reading about a horrific plane crash, so flying must be very dangerous."

Analysis Summary

The statement 'I remember reading about a horrific plane crash, so flying must be very dangerous' exhibits a cognitive bias known as the Availability Heuristic. This bias occurs when individuals assess the likelihood of an event based on how easily they can recall similar instances from memory. In this case, the vivid memory of a plane crash leads to an overestimation of the danger associated with flying.

Certainty Score

90%

Bias Detected

Bias Presents information with clear bias.

Detailed Analysis

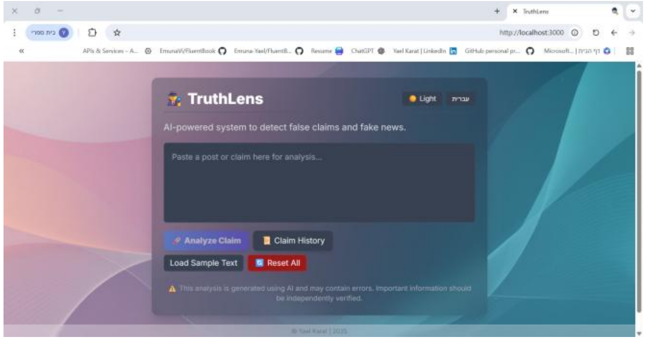
- Emotional language: 'horrific'
- The text itself provides the evidence of the individual's reliance on a single vivid memory to make a generalization about the safety of flying.

Technical Details

Timestamp: 8/6/2025, 2:32:56 PM
Model Used: gpt-3.5-turbo
Token Usage: 619

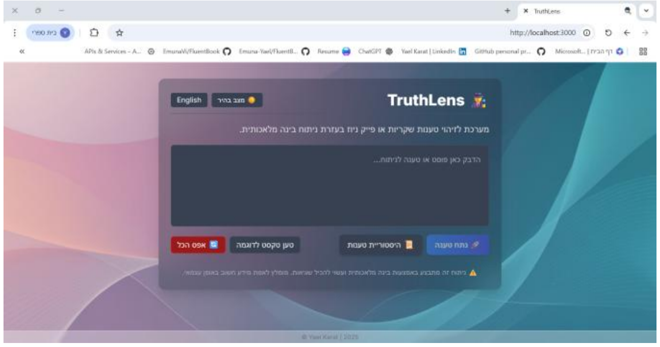
© Yael Karat | 2025

Dark Mode RTL Support:



AI-powered system to detect false claims and fake news.

This analysis is generated using AI and may contain errors. Important information should be independently verified.



מערכת לזיהוי טענות שקריות או פייק ניוז בעזרת ניתוח בינה מלאכותית.

ניתוח זה מתבצע באמצעות בינה מלאכותית ועשוי להכיל שגיאות. מומלץ לאמת מידע חשוב באופן עצמאי.

10. Conclusion

TruthLens demonstrates the potential of combining AI, UX design, and ethical responsibility to tackle misinformation and raise user awareness of biases. It has been a meaningful learning journey in NLP integration, multilingual UX, ethical computing, and cognitive science. With future improvements, it can become a robust educational and media-literacy tool.