FactFinder

Justice League

Agenda

- Team Introduction
- Improvements from professor feed back
- Minimum Viable Product
- Architecture diagrams
- Sequence diagrams
- Sprint 2 Recap
- Product Backlog
- Sprint 3 Backlog
- Metrics
- Retrospective
- Stories planned for sprint 4
- Project demo for sprint 3
- Live application demo

Meet our team

Suraj Salunke

Quality Analyst & Developer

Manthan Kale

Scrum Master & Developer

Kavita Kamtekar

Full Stack Developer

Saurabh Chaudhary

Product Manager & Developer

Gayatri Kulkarni

Developer

Maheswari Vidyadharani

Developer

Sai Kumar Tata

Developer & Desigining

Rushabh Shingala

Developer & Desigining

Improvements made from Professor Feedback

1. Followed the order of slides as per checklist
2. Updated User stories for product backlog
3. Improved Metrices slide
4. Improved Live application demo

Project Description

Fact Finder is a web app designed to combat online misinformation by differentiating between fake and real news. Additionally, it can create a summary of the news.
For anyone concerned about online misinformation and facing time constraints for reading news articles.
who upload the news article **the** Fact Finder web app **is a web app** that use machine learning and deep learning algorithms **that** can detect misinformation.
Unlike believe any information on internet.
Our application verifies the authenticity of information and saves time by creating summaries of articles.

Benefit Outcomes:

- Ensuring the authenticity of the information.
- Saves valuable time by generating summaries of the articles.
- Contributes to the fight against misinformation.

Team working agreement

Team Agreement

CS691 - Team Justice League (23915)

Purpose of the Agreement:

To ensure the successful completion of our collaborative project and foster a positive working environment for all team members.

Team Values:

1. Collaboration: We value open communication and collaboration, encouraging all team members to actively engage in discussions and contribute ideas for the project's success.
2. Accountability: Each team member is accountable for their assigned tasks, and if challenges arise, open communication is essential to address and overcome obstacles collectively.
3. Transparency: We prioritize transparent communication through various channels to build trust within the team. Keeping cameras on during Zoom meetings is encouraged for meaningful team interactions.

Communication:

1. The team will utilize Zoom for weekly meetings to facilitate meaningful discussions. Weekly meetings will be held every Tuesday, Friday, and Sunday. Attendance is mandatory, with exceptions allowed in exceptional cases.
2. Active participation in meetings is expected from every team member, including sharing ideas, engaging in discussions, and providing updates on individual work progress.
3. For immediate discussions, urgent matters, and doubts, a WhatsApp messenger group will be employed.
4. Microsoft Sharepoint will be the designated platform for sharing final deliverables, allowing all team members to collaboratively edit documents.
5. A shared platform, such as Jira, will be used for project management. It includes designated groups for different roles, facilitating efficient collaboration among Developers, Business Analysts, and the Product Owner.

Work Division and Participation:

1. Project work will be equitably divided among team members, with equal responsibilities assigned to ensure a balanced workload.
2. Timely completion of assigned work is crucial. In cases of potential delays, team members must communicate with their peers to redistribute tasks accordingly.
3. Work separation between members is voluntary. However, if a member lacks participation, the Product Owner reserves the right to assign necessary tasks to ensure project progress.
4. In the event of a member's absence during meetings, the member pledges to support the decisions made during the meeting.

Team Members:

1. Saurabh chaudhary
2. Suraj Salunkhe
3. Kavita Kamtekar
4. Manthan Kale
5. Maheswari Vidyadharani
6. Rushabh Shingala
7. Sai Kumar Tata
8. Gayatri Kishor Kulkarni

Personas

Background: Maria is a community activist working on social justice issues. She recognizes the role of misinformation in shaping public opinion and wants to combat false narratives that may undermine her advocacy efforts. Maria is motivated to use the web application as a tool to verify information before sharing it within her community.

Age: 35

Occupation: Community Activist

Goals and Motivation: Maria's goal is to empower her community with reliable information. By leveraging the web application, she aims to strengthen the credibility of her advocacy work, foster informed discussions, and counteract misinformation that may be used to undermine social justice causes

Personas

Background: David owns a small business that heavily relies on its online presence for customer engagement. He's concerned about potential misinformation impacting the reputation of his business. The web application is crucial for David to verify news and updates related to his industry.

Age: 38

Occupation: Small Business Owner

Goals and Motivation: David's primary goal is to safeguard his business's reputation. By using the web application, he aims to prevent the spread of false information that could harm customer trust and loyalty. His motivation is to maintain transparency and integrity in his business communications.

Personas

Background: Michelle is a fitness and health enthusiast who frequently relies on online health-related information. With the abundance of health-related news articles and social media posts, Michelle often encounters conflicting information. The web application is valuable for him to discern between credible and misleading health information.

Age: 31

Occupation: Fitness Trainer

Goals and Motivation: Michelle's primary goal is to maintain a healthy lifestyle and share accurate health advice with her followers. By using the web application, he aims to avoid the spread of misinformation in the fitness and health community, contributing to a more informed and health-conscious online environment.

MVP(Minimum Valuable Product):

1) The user will also be able to check news without having Account in FactFinder unless user wants to check his search News History.
2) The "Homepage" tab will display a short video of what the Fake news is.
3) The user should navigate to "Verify News" tab and post news as a text in the Textbox.
4) The user will click on verify news button.
5) The system will then generate a response based on our ML model.
6) The user will be able to get a response if the news/text posted was Fake or Real.
7) The "About us" tab will display explanation of the website, what it does and how it works.
8) The user can Register with email-ID and password and user should able to click on submit button to get register successfully.
9) The user should Login with email-ID and password.
10)Users can chat with like-minded people and discuss about any latest news for more information.

Technologies

1. FrontEnd: JavaScript, HTML5, CSS.
2. Framework: React, Flask.
3. Backend : Python
4. Database : NoSQL, Firebase

Algorithms

Machine Learning Algorithms (scikit-Learn)

Logistic Regression : Supervised learning algorithm for classification.

Recurrent Neural Network(RNN): Ideal for Natural Language Processing(NLP) task.

Algorithms(cont..)

```
import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import  LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
import re
import string
```

```
# creating another method to process the text
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

port_stem = PorterStemmer()

def clean_and_lower(text):
    cleaned_text = re.sub(r'[^A-Za-z0-9]+', ' ', text)
    cleaned_text = cleaned_text.lower()
    cleaned_text = cleaned_text.split()
    cleaned_text = [port_stem.stem(word) for word in cleaned_text if not word in stopwords.words('english')]
    cleaned_text = ' '.join(cleaned_text)

    return cleaned_text
```

Algorithms(cont..)

```
df['content'] = df['content'].apply(clean_and_lower)
```

```
X = df['content']
y = df['classification']
```

```
X_train, X_test, y_train,y_test = train_test_split(X,y,test_size=0.25,stratify=y,random_state=42)
X_train = vectorization.fit_transform(X_train)
X_test = vectorization.transform(X_test)
```

```
print(y_train.value_counts())
print(y_test.value_counts())
```

```
1    17610
0    16063
Name: classification, dtype: int64
1    5871
0    5354
Name: classification, dtype: int64
```

```
Logistic_model = LogisticRegression()
Logistic_model.fit(X_train,y_train)
Logistic_model.score(X_test,y_test)
```

```
]: 0.9865478841870824
```

Algorithms(cont..)

```
X = df_main['Text']
y = df_main['summary']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state = 42)


#Tokenize input and target summaries
tokenizer_x = Tokenizer(num_words=1000)
tokenizer_x.fit_on_texts(X_train)
X_train_sequences = tokenizer_x.texts_to_sequences(X_train)
#max_input_len = max(len(seq) for seq in X_train_sequences)

tokenizer_y = Tokenizer(num_words=250)
tokenizer_y.fit_on_texts(y_train)
y_train_sequences = tokenizer_y.texts_to_sequences(y_train)
#max_traget_len = max(len(seq) for seq in y_train_sequences)

X_train_padded = pad_sequences(X_train_sequences, maxlen = 1000, padding = 'post')
y_train_padded = pad_sequences(y_train_sequences, maxlen = 250, padding = 'post')


#encoder
encoder_input = Input(shape = (1000,))
encoder_embedding = Embedding(input_dim = 1000, output_dim = 250)(encoder_input)
encoder_lstm = LSTM(256,return_sequences=True, return_state= True)
encoder_output, state_h, state_c = encoder_lstm(encoder_embedding)
encoder_states = [state_h, state_c]

#decoder

decoder_input_data = pad_sequences(y_train_sequences, maxlen = 250, padding = 'post')[:,:-1]
decoder_target_data = pad_sequences(y_train_sequences, maxlen = 250, padding = 'post')[:,1:]

decoder_input = Input(shape=(None,))
decoder_embedding = Embedding(input_dim = 1000, output_dim = 250)(decoder_input)
decoder_lstm = LSTM(256,return_sequences=True, return_state= True)
decoder_output, _, _ = decoder_lstm(decoder_embedding, initial_state = encoder_states)
attention = Attention()
context_vector = attention([decoder_output,encoder_output])
decoder_combined_context = tf.concat([decoder_output, context_vector], axis=-1)
decoder_dense = Dense(250, activation = 'softmax')
decoder_output = decoder_dense(decoder_combined_context)

RNN_model = Model([encoder_input,decoder_input], decoder_output)
RNN_model.compile(optimizer = 'adam', loss= 'sparse_categorical_crossentropy')
RNN_model.fit([X_train_padded,decoder_input_data], decoder_target_data, batch_size=32, epochs = 10)
```

Algorithms(cont..)

In [46]:

```
# Preprocess test data
X_test_sequences = tokenizer_x.texts_to_sequences(X_test)
X_test_padded = pad_sequences(X_test_sequences, maxlen=1000, padding='post')

y_test_sequences = tokenizer_y.texts_to_sequences(y_test)
decoder_input_test_data = pad_sequences(y_test_sequences, maxlen=250, padding='post')[:, :-1]
decoder_target_test_data = pad_sequences(y_test_sequences, maxlen=max_traget_len, padding='post')[:, 1:]

# Generate predictions on test data
predicted_summaries = RNN_model.predict([X_test_padded, decoder_input_test_data], batch_size=32)
```

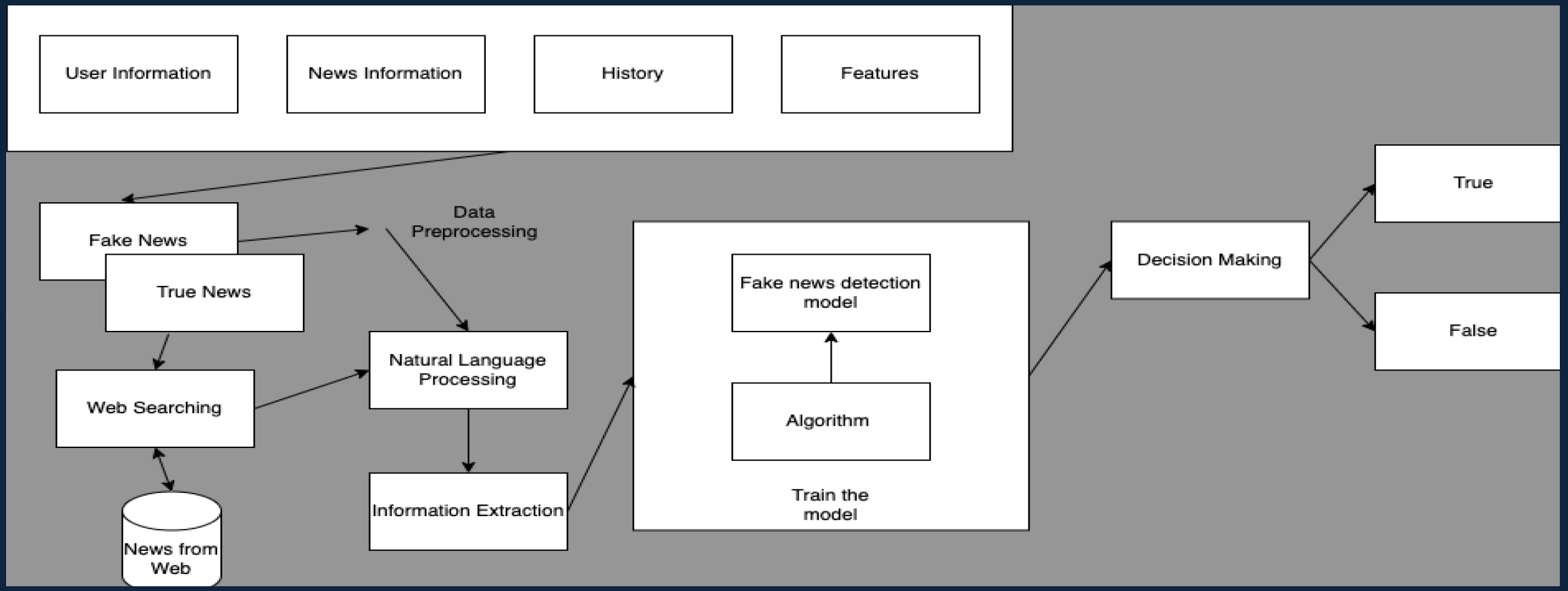
```
18/18 [==============================] - 29s 2s/step
```

In [ ]:

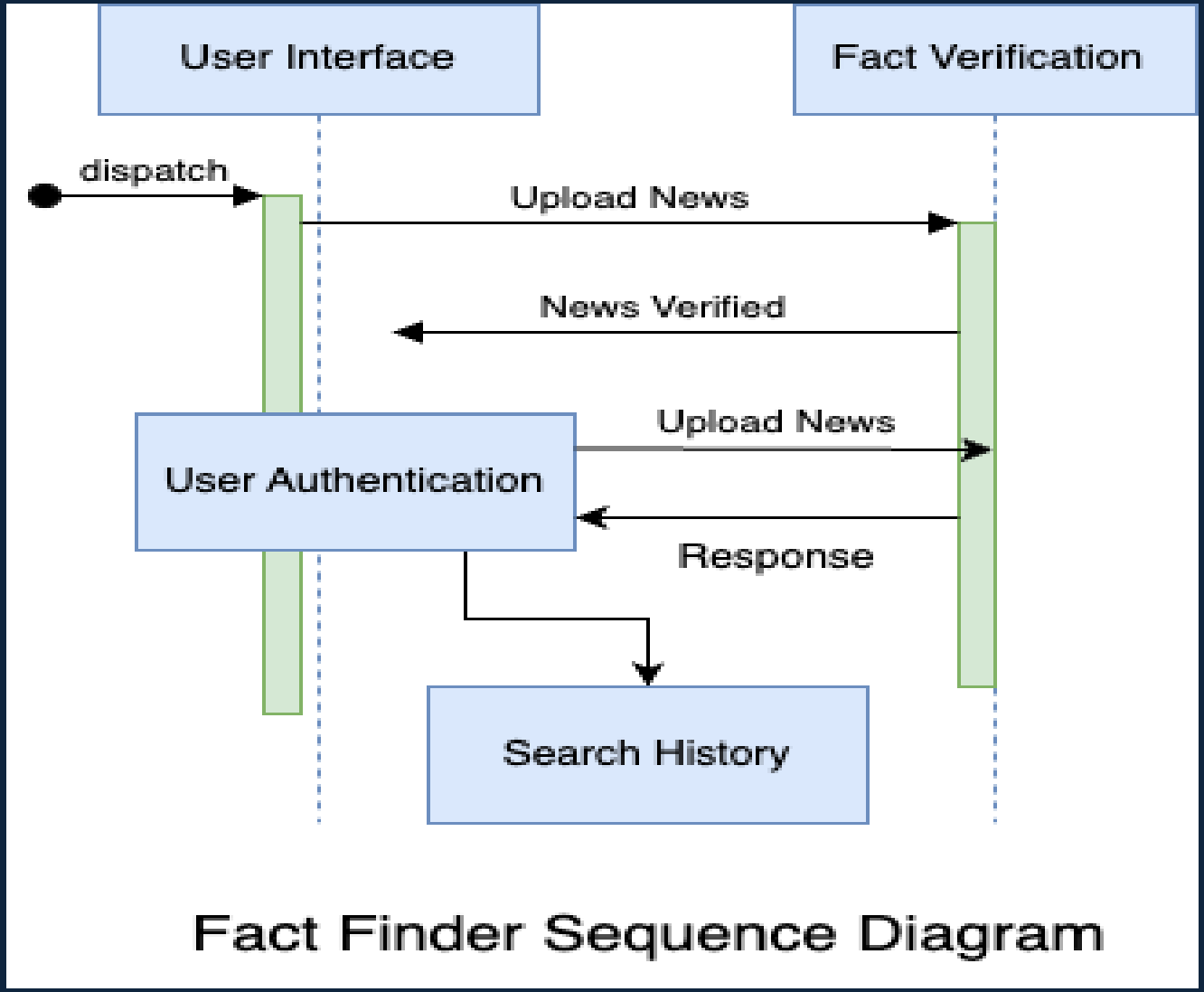
In [48]:

```
import numpy as np
#decoding predicted summaries from sequences to text
predicted_summaries_text = []
for summary_sequence in predicted_summaries:
    predicted_summary = []
    for token_index in summary_sequence:
        predicted_word = tokenizer_y.index_word.get(np.argmax(token_index))
        if predicted_word is None or predicted_word == '<end>':
            break
        predicted_summary.append(predicted_word)
    predicted_summary_text = ' '.join(predicted_summary)
    predicted_summaries_text.append(predicted_summary_text)

#evaluating the predicted summary to actual summary

for i in range(5):
    print("Input Text:", X_test.iloc[i])
    print()
    print("Target Summary:", y_test.iloc[i])
    print()
    print("Predicted Summary:", predicted_summaries_text[i])
    print()
```

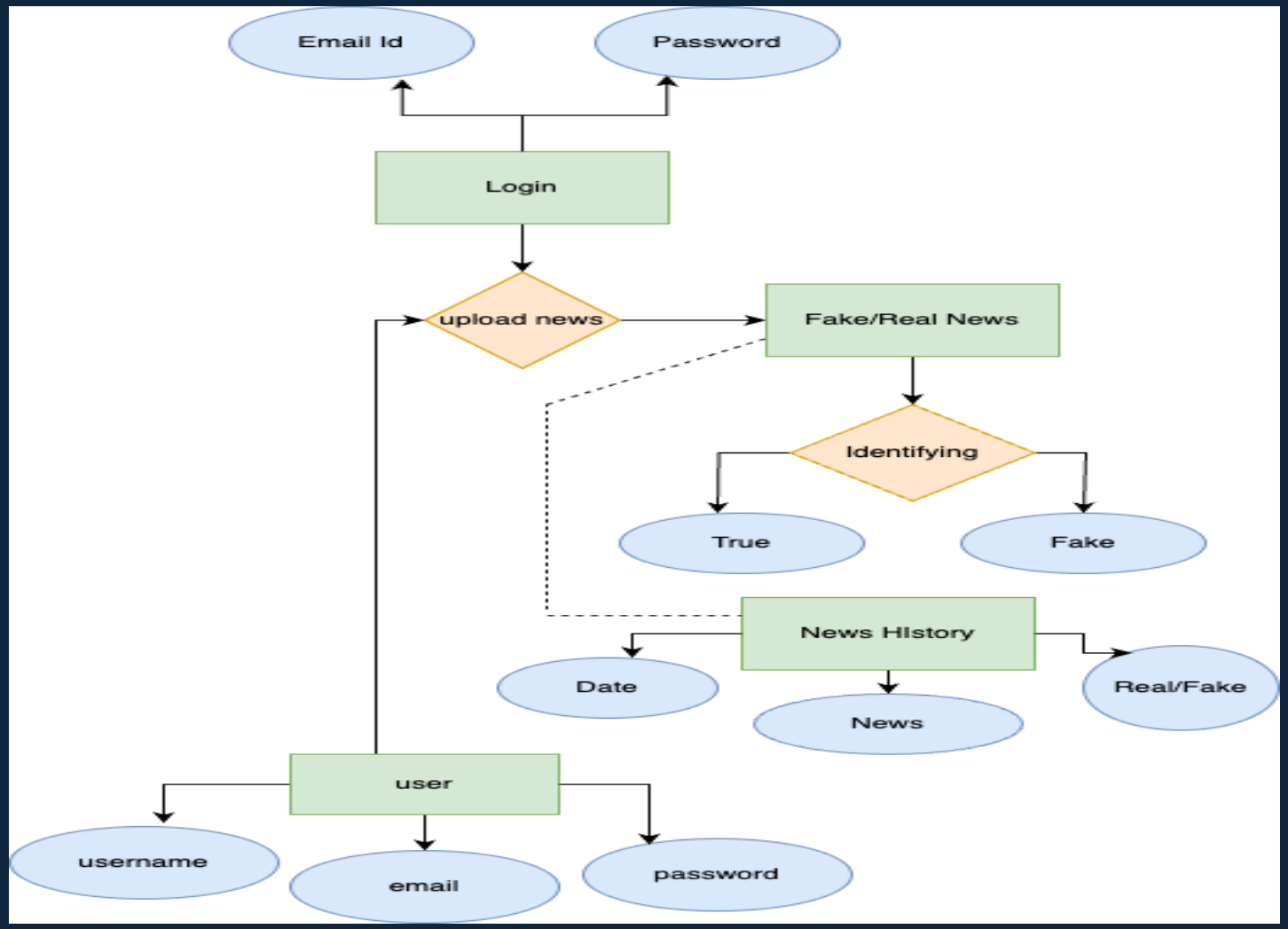
Architecture Diagram



Sequence Diagram



Entity-Relation Diagram:



Sprint 2 Recap

- Developed ML model.
- Created UI Design (Login and Home page, Search Textbar).
- Created Architecture, ER and Sequence Diagram.
- Created backend to train ML model.
- Tested Backend and frontend functionality.

Product Backlog

Sprint 2

User Story ID	User stories	Acceptance Criteria
FNDJL-4	As a registered user, I want to be able to log in to the application so that I can access the website.	1. Given the login page is displayed, when I enter valid credentials and click the "Login" button, then I should be redirected to the home page. 2. Given the login page is displayed, when I enter an invalid username and valid password, then an error message should be shown indicating that the username is not recognized. 3. Given the login page is displayed, when I enter a valid username and an invalid password, then an error message should be displayed indicating that the password is incorrect. 4. Given the login page is displayed, when I leave both username and password fields blank and click the "Login" button, then an error message should be shown indicating that both fields are required.
FNDJL-5	As a security measure, I want my account to be locked after a certain number of unsuccessful login attempts.	1.Given a user has made multiple unsuccessful login attempts, when the maximum allowed attempts are reached, then the user account should be locked. 2. Given a locked account, when the user tries to log in, then an error message should be displayed indicating that the account is locked. 3. Given a locked account, when the user clicks on a "Forgot Password" link, then they should be directed to a password recovery process.

Product Backlog

Sprint 2

User Story ID	User stories	Acceptance Criteria
JL-6	As a new user, I want to register on the website to access exclusive features and personalized content.	Scenario: Successful Registration Given that I am on the registration page, When I enter valid information (first name, last name, email address, and password), And click on the "Register" button, Then I should receive a confirmation message indicating successful registration. Scenario: Invalid Email Address Given that I am on the registration page, When I enter an invalid email address (e.g., without "@" symbol), And click on the "Register" button, Then I should see an error message indicating that the email address is invalid.
FNDJL-9	As a user, I want to learn more about the application and its creators to understand its purpose and credibility.	Scenario: Accessing About Us Page Given: I am on the home page. When: I navigate to the "About Us" tab. Then: The system should display information about the purpose of the application, its creators, and any other relevant details.
FNDJL-10	As a user, I want to easily find and access contact information to reach out for support or inquiries.	Scenario: Accessing Contact Page Given: I am on any tab other than "Contact." When: I navigate to the "Contact" tab. Then: The system should display contact information or a form for users to get in touch.

Product Backlog

Sprint 3

User Story ID	User stories	Acceptance Criteria
FNDJL-11	As a user, I want to have access to a user-friendly interface where I can easily input news content for fact-checking.	The input text box should be prominently displayed on the homepage or main landing page. The interface should be intuitive and easy to understand for users of all backgrounds. Users should receive clear feedback after submitting news content for fact-checking.
FNDJL-12	As a user, I want to receive fact-checking results in a timely manner, without significant delay.	The fact-checking process should be optimized for speed and efficiency. Users should receive results within a reasonable timeframe after submitting news content. The website should indicate if there are any delays due to high traffic or other technical issues.
FNDJL-13	As a user, I want to be able to view detailed analysis and sources for fact-checked news articles.	Along with the determination of whether the news is fake or true, users should be provided with detailed analysis and sources. The analysis should highlight specific reasons for the determination, such as misleading information or lack of credible sources. Sources should be clickable, allowing users to verify the information independently.
FNDJL-14	As a user, I want to be able to view my search history.	There should be a dedicated section or page on the website where I can view my search history. The search history should be displayed in a clear and organized manner, showing the date and time of each search. Each item in the search history should be clickable, allowing me to view the details of that particular search. I should have the option to clear my search history if desired. The search history should be stored securely and only accessible to the user who performed the searches.
FNDJL-15	As a user, I want to have access to a feedback mechanism to report misinformation or suggest corrections for fact-checked news articles.	Users should be able to easily report misinformation or suggest corrections for fact-checked news articles. There should be a clear and accessible feedback button or form on each news article page. Users should receive acknowledgment of their feedback and be informed of any actions taken in response.

Product Backlog

Sprint 4

User Story ID	User storiesAcceptance Criteria	Acceptance Criteria
FNDJL-16	As a user, I want to have the option to share fact-checked news articles on social media platforms.	Users should be able to easily share fact-checked news articles via popular social media platforms like Facebook, Twitter, and LinkedIn. Sharing options should be prominently displayed on the website interface. Shared posts should include a brief summary and link back to the original fact-checked article on the website.
FNDJL-17	As a user, I want to receive notifications or alerts for breaking news stories that are being fact-checkedU	Users should have the option to opt-in to receive notifications for breaking news stories. Notifications should be delivered in a timely manner via email or push notifications. Users should be able to customize their notification preferences, such as frequency and types of stories.
FNDJL-18	As a user, I want to have the option to filter and sort my search history for better organization and retrieval.	Users should be able to filter search history by date range, keyword, or other relevant criteria. Search history should support sorting options such as by date, relevance, or alphabetical order. Filtering and sorting options should be intuitive and easy to use.
FNDJL-19	As a user, I want to have the option to engage with a community of fellow users to discuss news topics, share insights, and collaborate on fact-checking efforts.	The website should include a community forum or discussion board where users can create posts, comment on threads, and interact with each other. Users should be able to join specific topic-based groups or communities within the platform. Moderation tools should be in place to ensure a respectful and constructive environment for discussions.

Sprint 3 backlog

Fact Finder
accuracy, integrity and fairness

User Story ID	User stories	Acceptance Criteria	Status
FNDJL-11	As a user, I want to have access to a user-friendly interface where I can easily input news content for fact-checking.	The input text box should be prominently displayed on the homepage or main landing page. The interface should be intuitive and easy to understand for users of all backgrounds. Users should receive clear feedback after submitting news content for fact-checking.	Completed
FNDJL-12	As a user, I want to receive fact-checking results in a timely manner, without significant delay.	The fact-checking process should be optimized for speed and efficiency. Users should receive results within a reasonable timeframe after submitting news content. The website should indicate if there are any delays due to high traffic or other technical issues.	Completed
FNDJL-13	As a user, I want to be able to view detailed analysis and sources for fact-checked news articles.	Along with the determination of whether the news is fake or true, users should be provided with detailed analysis and sources. The analysis should highlight specific reasons for the determination, such as misleading information or lack of credible sources. Sources should be clickable, allowing users to verify the information independently.	Completed
FNDJL-14	As a user, I want to be able to view my search history.	There should be a dedicated section or page on the website where I can view my search history. The search history should be displayed in a clear and organized manner, showing the date and time of each search. Each item in the search history should be clickable, allowing me to view the details of that particular search. I should have the option to clear my search history if desired. The search history should be stored securely and only accessible to the user who performed the searches.	Completed
FNDJL-15	As a user, I want to have access to a feedback mechanism to report misinformation or suggest corrections for fact-checked news articles.	Users should be able to easily report misinformation or suggest corrections for fact-checked news articles. There should be a clear and accessible feedback button or form on each news article page. Users should receive acknowledgment of their feedback and be informed of any actions taken in response.	Not Completed

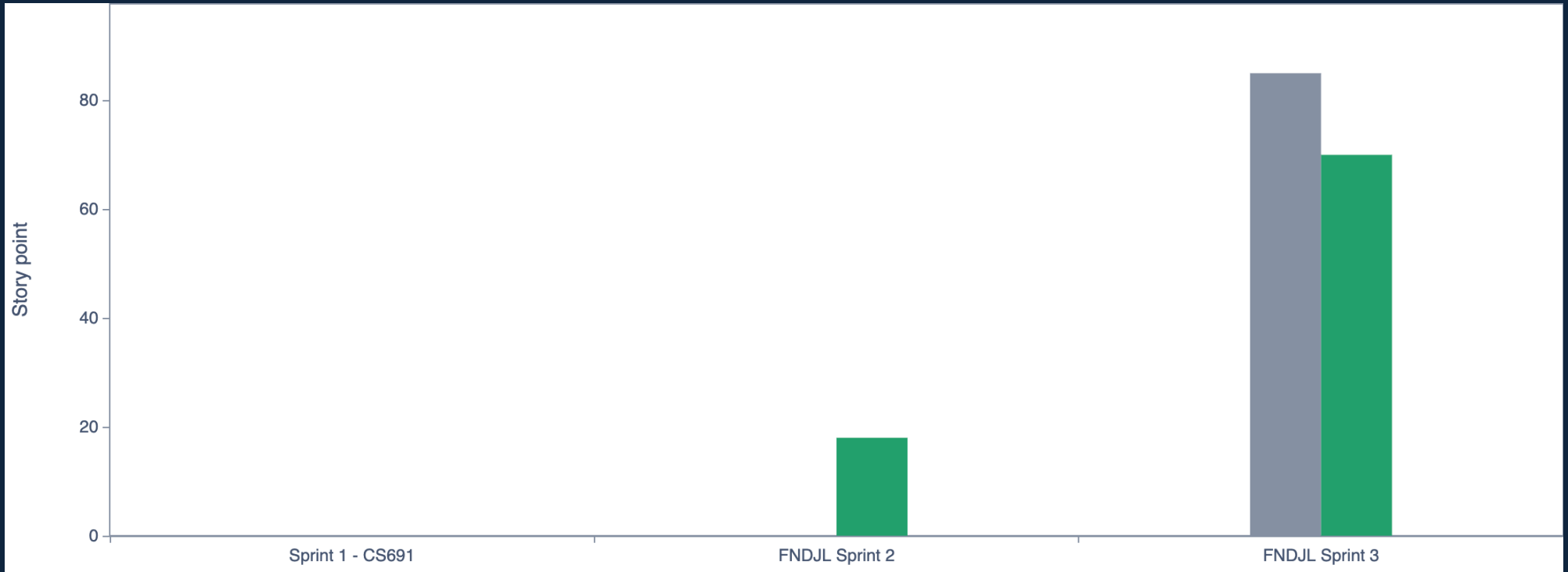
Test Cases

Fact Finder
accuracy, integrity and fairness

User Story ID	Test Case ID	Name	Objective	Expected Result	Test Data
FNDJL-11	FNDJL-T17	Submit Valid News Content	Verify the system accepts valid news content for fact-checking.	The system accepts the news content for processing. User receives a confirmation message	News Content: A short news article from a reputable source
FNDJL-11	FNDJL-T18	Empty News Content Submission	Verify the system rejects submissions with empty news content.	An error message is displayed indicating news content is required.	News Content: Empty text box.
FNDJL-11	FNDJL-T19	Large Text Submission	Verify the system handles submissions exceeding a character limit (if applicable).	An error message is displayed indicating the content is too long. (Optional: suggest truncation or provide a character count).	News Content: A lengthy news article exceeding the character limit (define
FNDJL-12	FNDJL-T20	Detailed Analysis	Verify the system displays detailed analysis alongside the truth determination for a fact-checked news article.	The fact-checking result is displayed (e.g., True or False). Alongside the result, detailed analysis is presented explaining the reasoning behind the determination. This may include: Highlighting specific phrases or information	N/A (This test uses previously submitted content)
FNDJL-12	FNDJL-T21	Missing Analysis for Fact-Checked Article	Verify the system handles cases where detailed analysis is unavailable for a fact-checked article.	The fact-checking result is displayed. A message is displayed indicating detailed analysis is unavailable due to insufficient information. (Optional: Offer alternative	N/A (This test uses previously submitted content)
FNDJL-14	FNDJL-T22	Search History Display	Verify a user's search history is displayed after submitting news content for fact-checking.	Upon successful submission, the user is redirected to their dashboard. The dashboard displays a dedicated section or page showcasing the user's search history. Each entry displays the date and time of the	N/A (This test uses previously submitted content)
FNDJL-14	FNDJL-T23	Search History Filtiring	Search History Verify users can filter their search history based on specific criteria.	The search history interface provides options to filter entries by various criteria (e.g., date range, keyword). Applying filters successfully narrows down the	N/A (This test uses previously submitted content)

Metrics

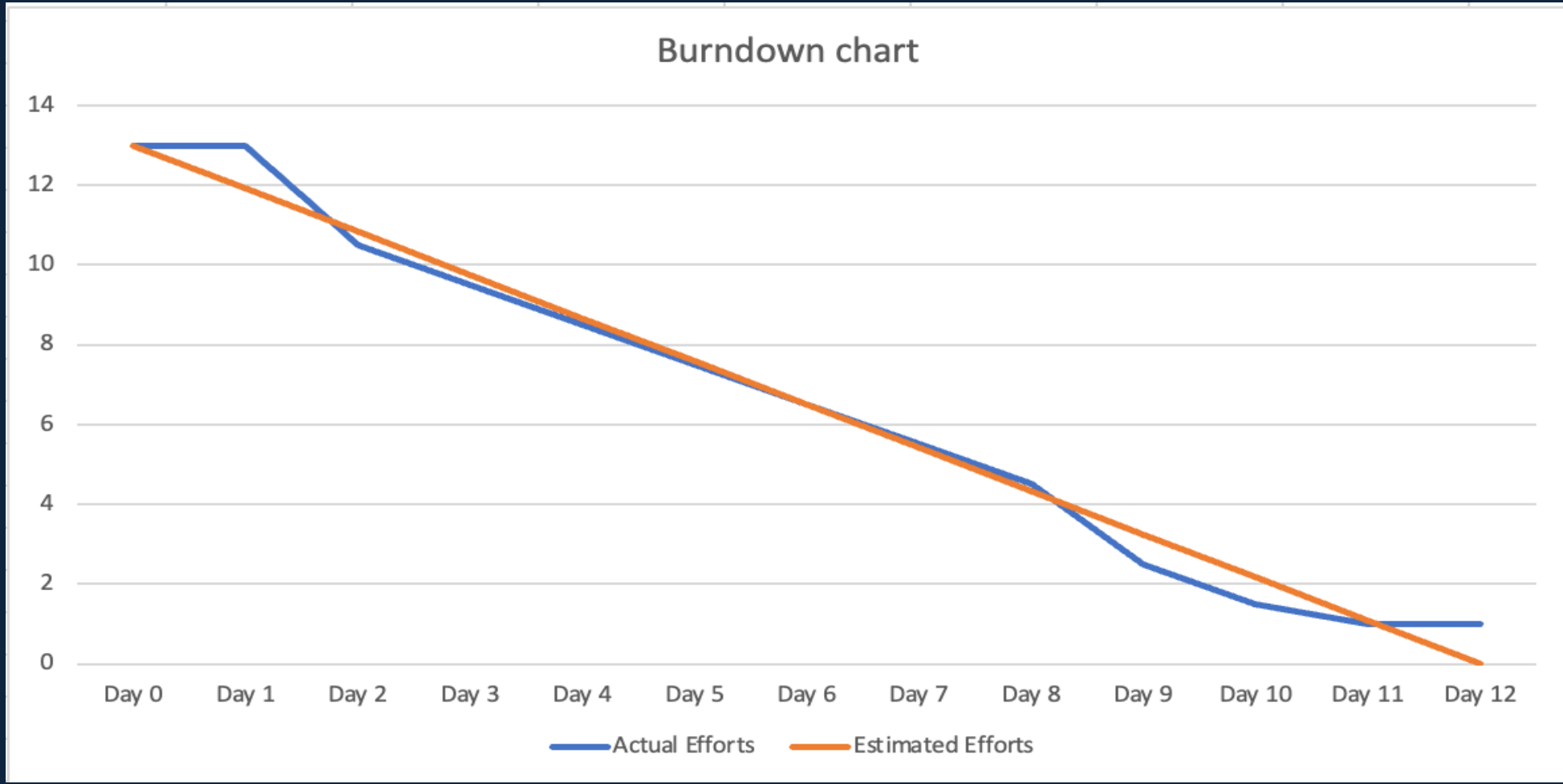
Team Velocity



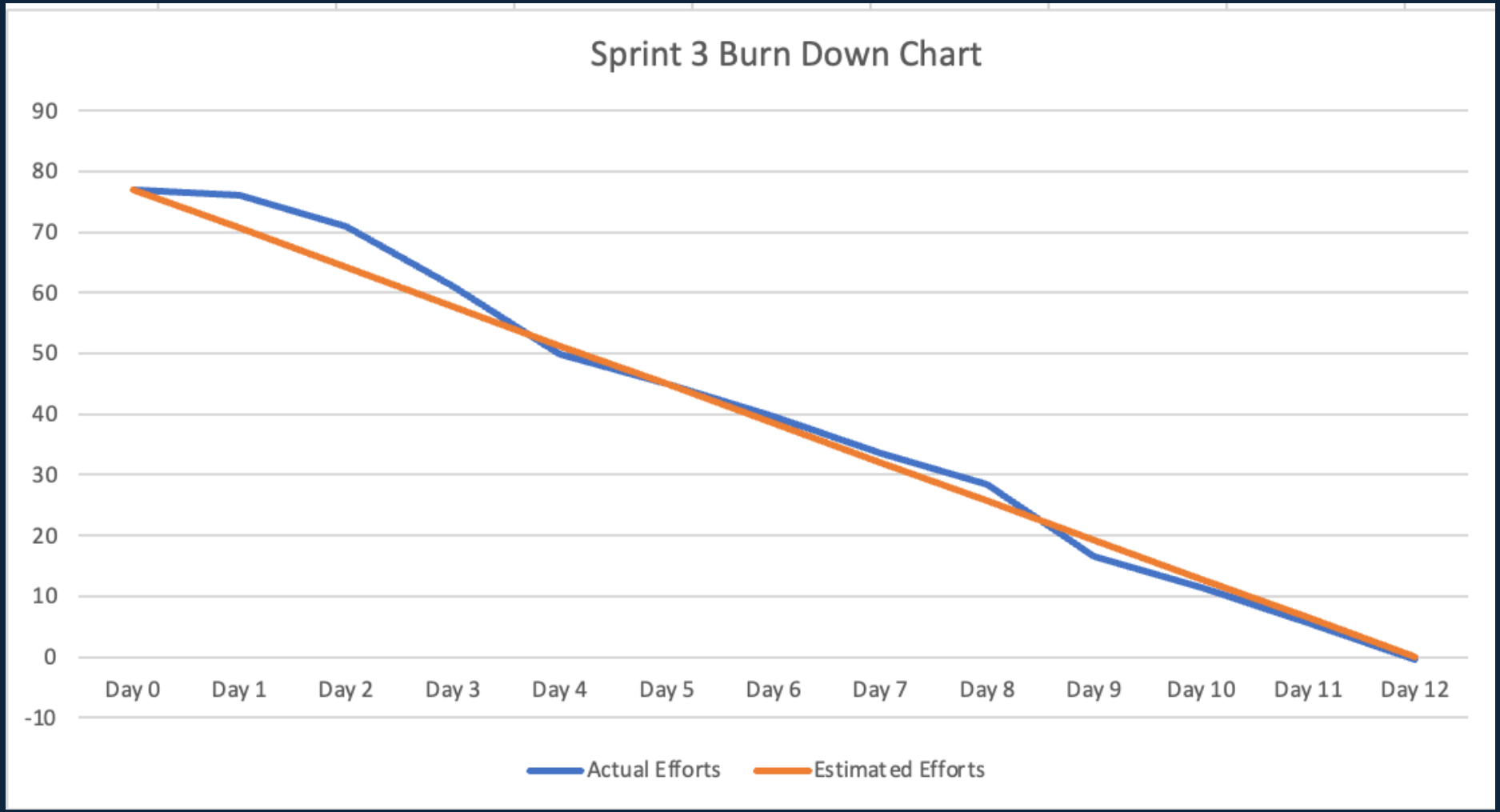
Sprint	Commitment	Completed
Sprint 1 - CS691	0	0
FNDJL Sprint 2	0	18
FNDJL Sprint 3	85	70

Completed/Commited Ratio : 82.35 %

Sprint 2 Burndown Chart

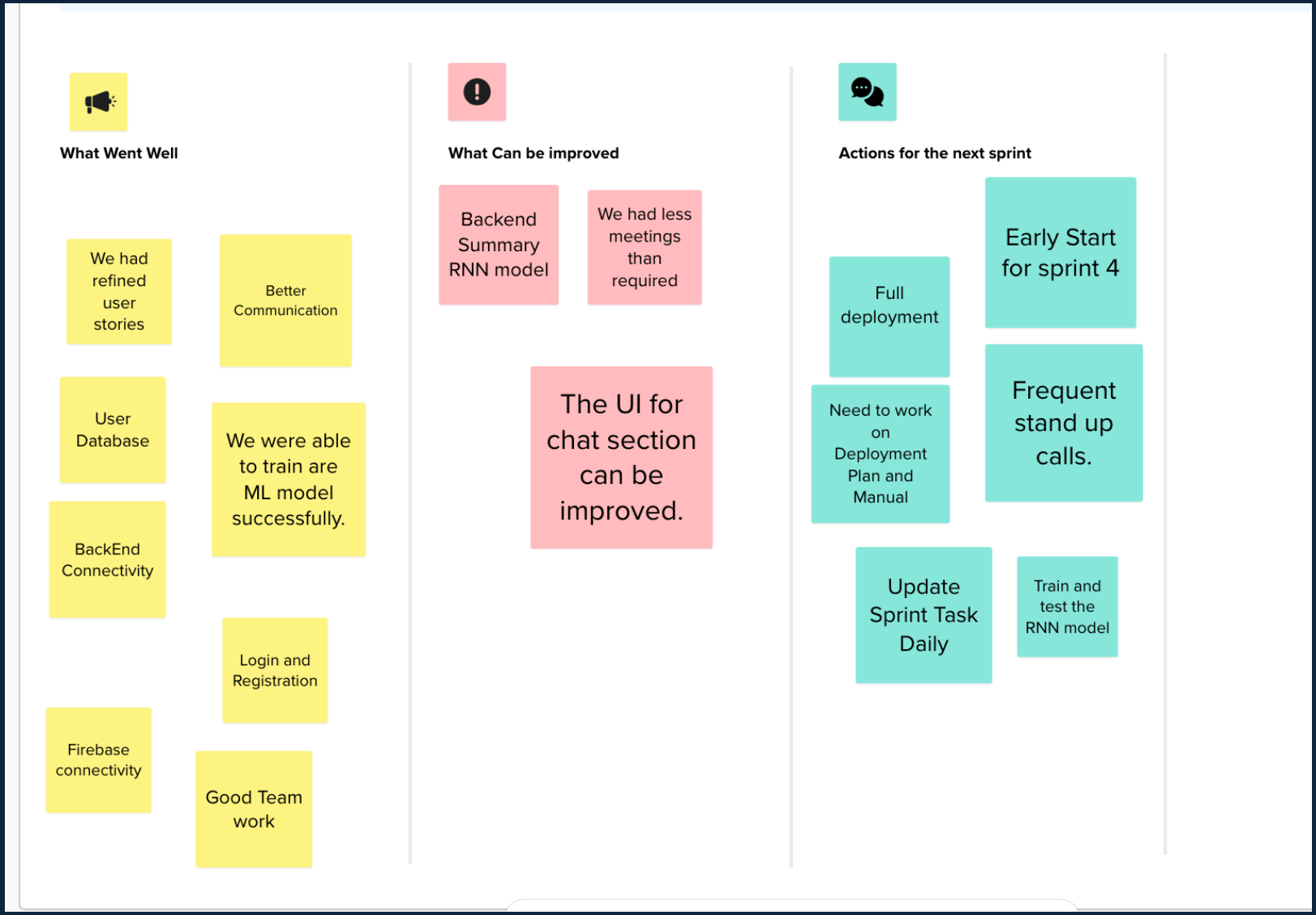


Sprint 3 Burndown Chart



Retrospective

Fact Finder
accuracy, integrity and fairness

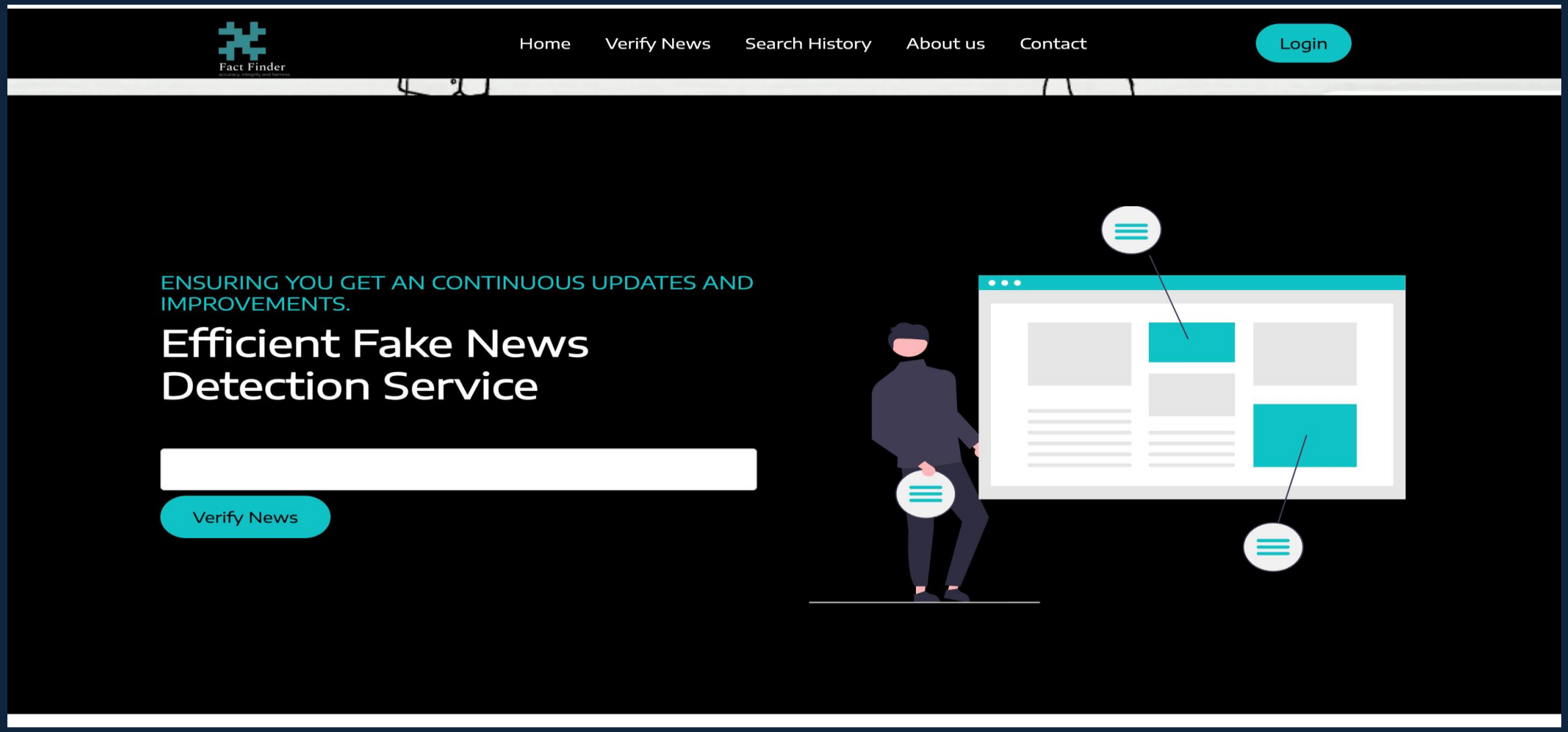


Stories and Acceptance criteria for Sprint 4

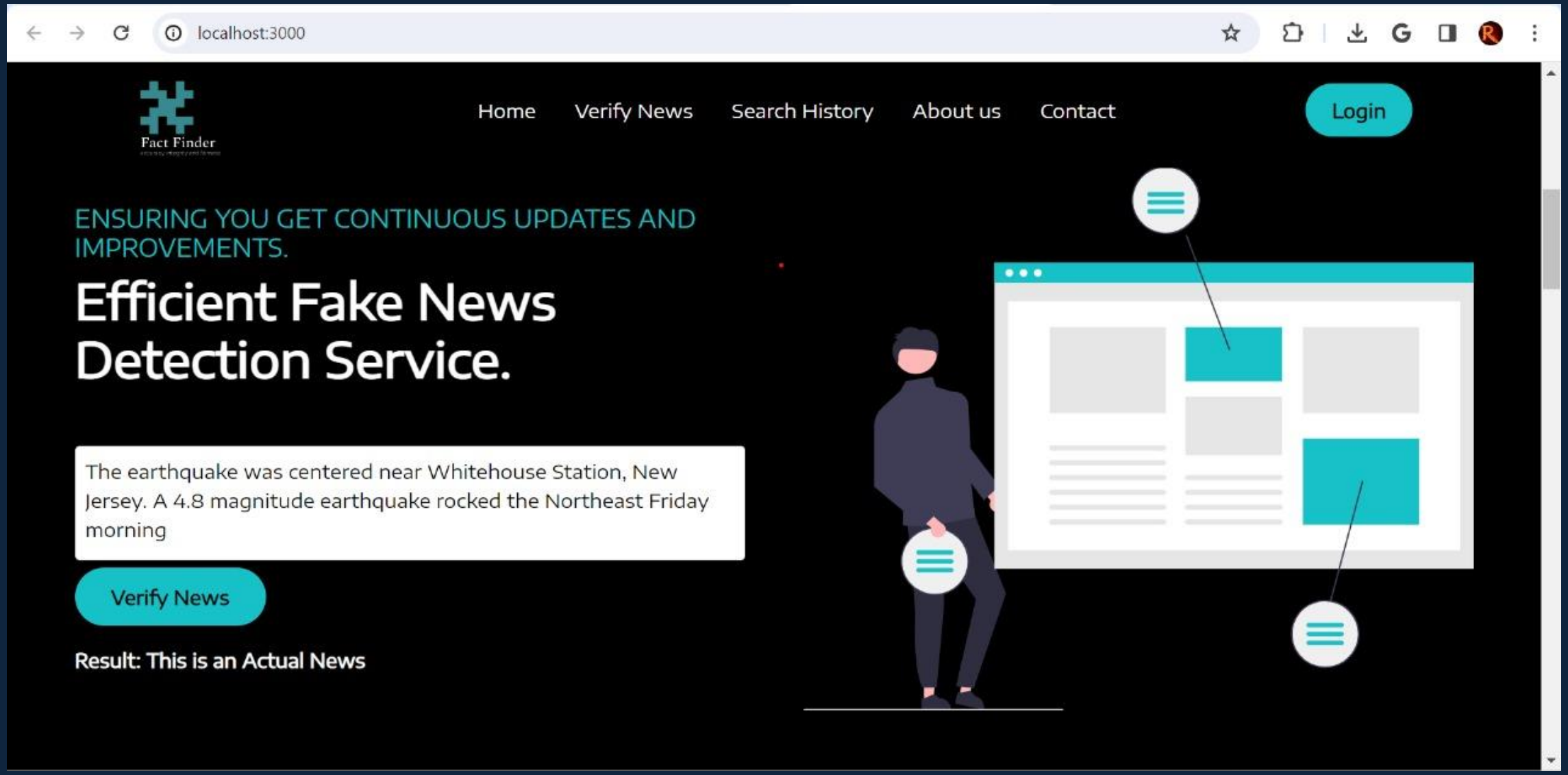
User Story ID	User storiesAcceptance Criteria	Acceptance Criteria
FNDJL-16	As a user, I want to have the option to share fact-checked news articles on social media platforms.	Users should be able to easily share fact-checked news articles via popular social media platforms like Facebook, Twitter, and LinkedIn. Sharing options should be prominently displayed on the website interface. Shared posts should include a brief summary and link back to the original fact-checked article on the website.
FNDJL-17	As a user, I want to receive notifications or alerts for breaking news stories that are being fact-checkedU	Users should have the option to opt-in to receive notifications for breaking news stories. Notifications should be delivered in a timely manner via email or push notifications. Users should be able to customize their notification preferences, such as frequency and types of stories.
FNDJL-18	As a user, I want to have the option to filter and sort my search history for better organization and retrieval.	Users should be able to filter search history by date range, keyword, or other relevant criteria. Search history should support sorting options such as by date, relevance, or alphabetical order. Filtering and sorting options should be intuitive and easy to use.
FNDJL-19	As a user, I want to have the option to engage with a community of fellow users to discuss news topics, share insights, and collaborate on fact-checking efforts.	The website should include a community forum or discussion board where users can create posts, comment on threads, and interact with each other. Users should be able to join specific topic-based groups or communities within the platform. Moderation tools should be in place to ensure a respectful and constructive environment for discussions.

FactFinder website screenshots:

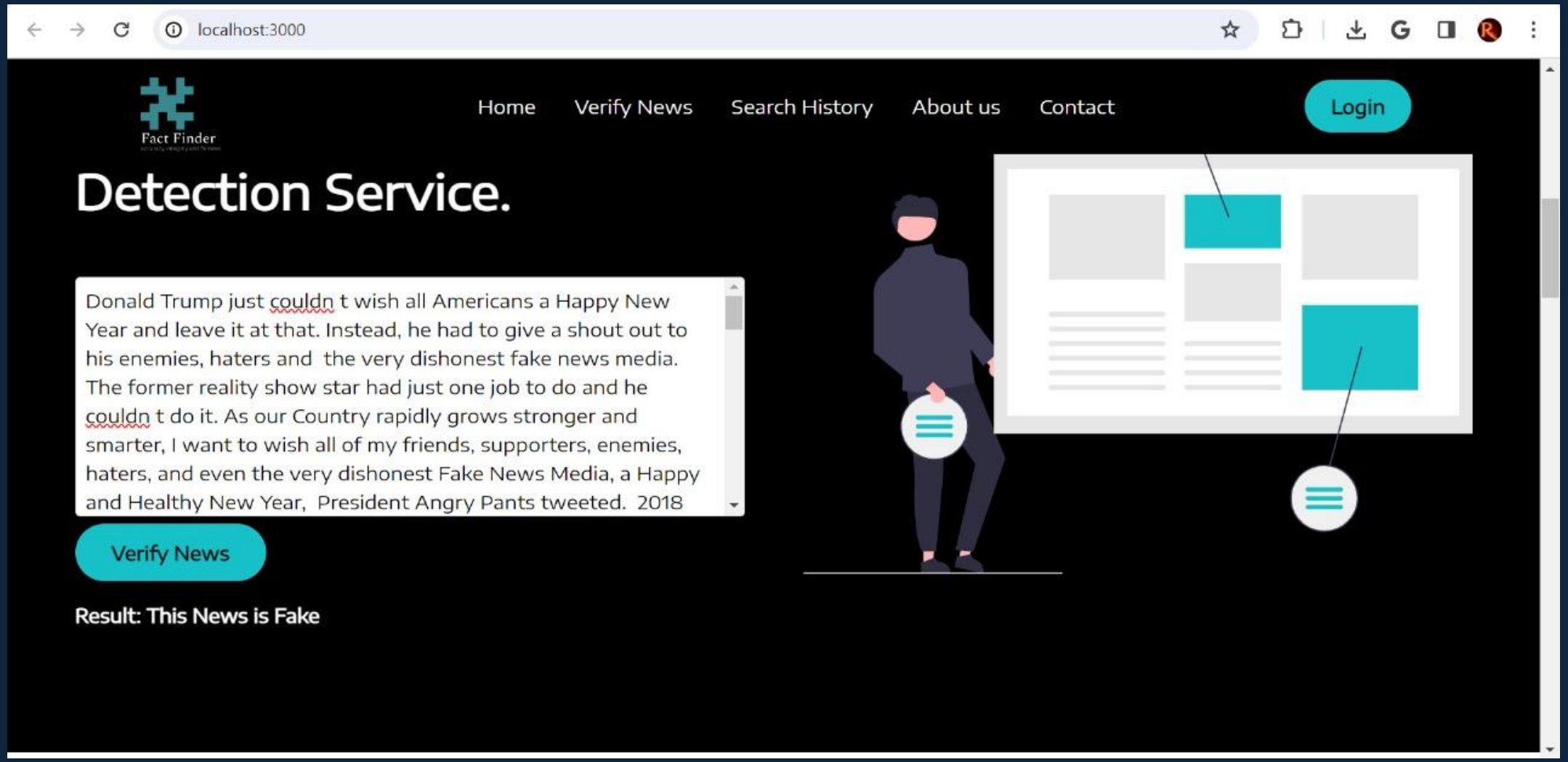
FactFinder website screenshots:



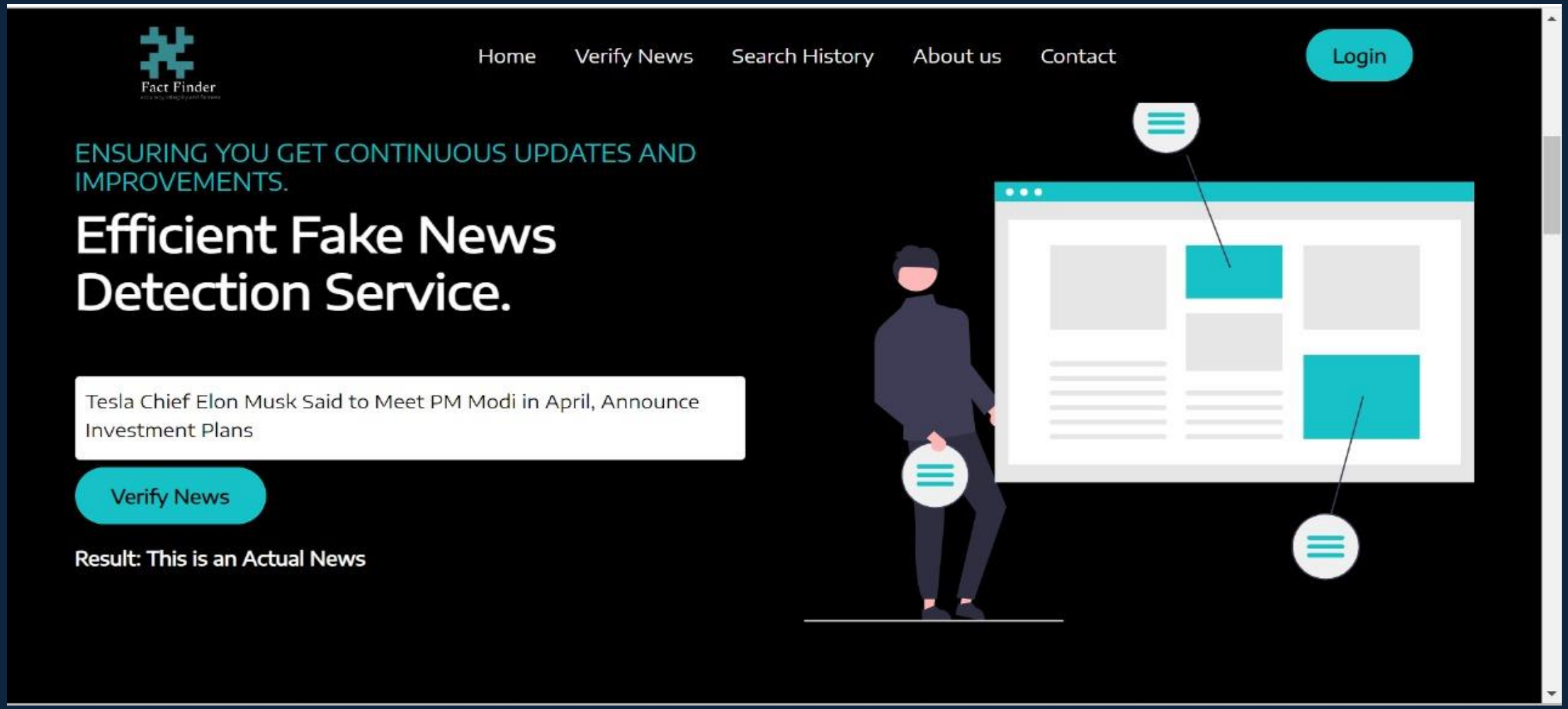
FactFinder website screenshots:



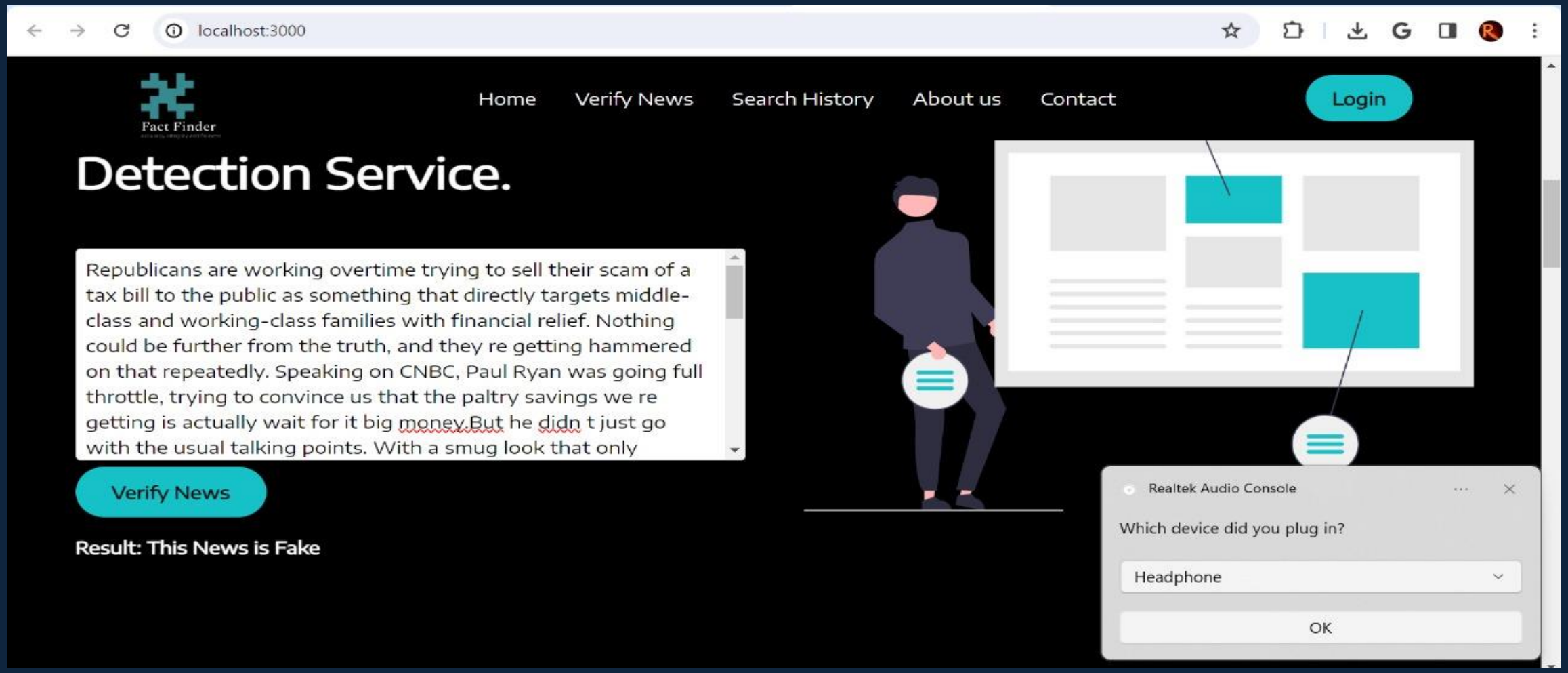
FactFinder website screenshots:



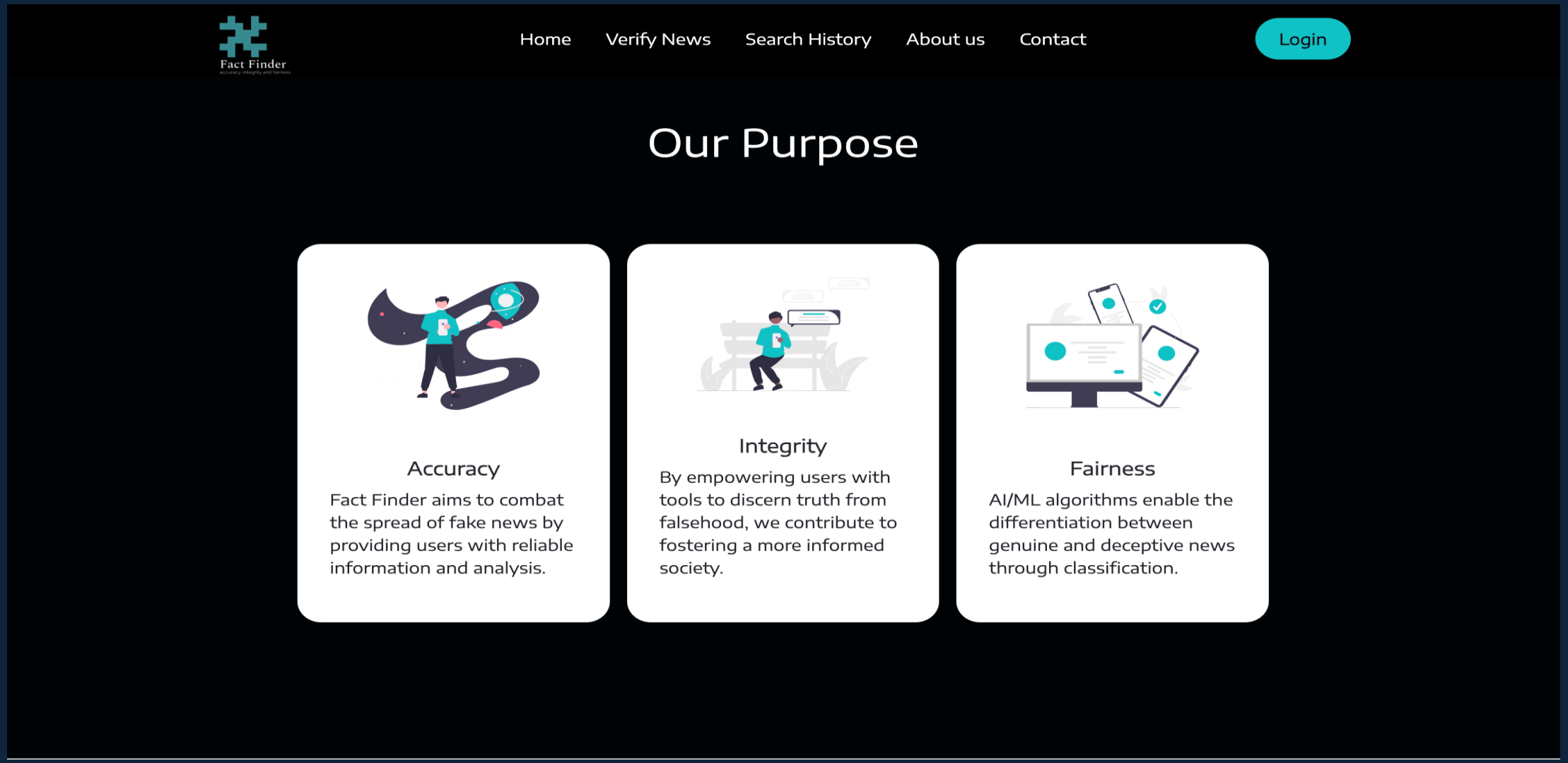
FactFinder website screenshots:



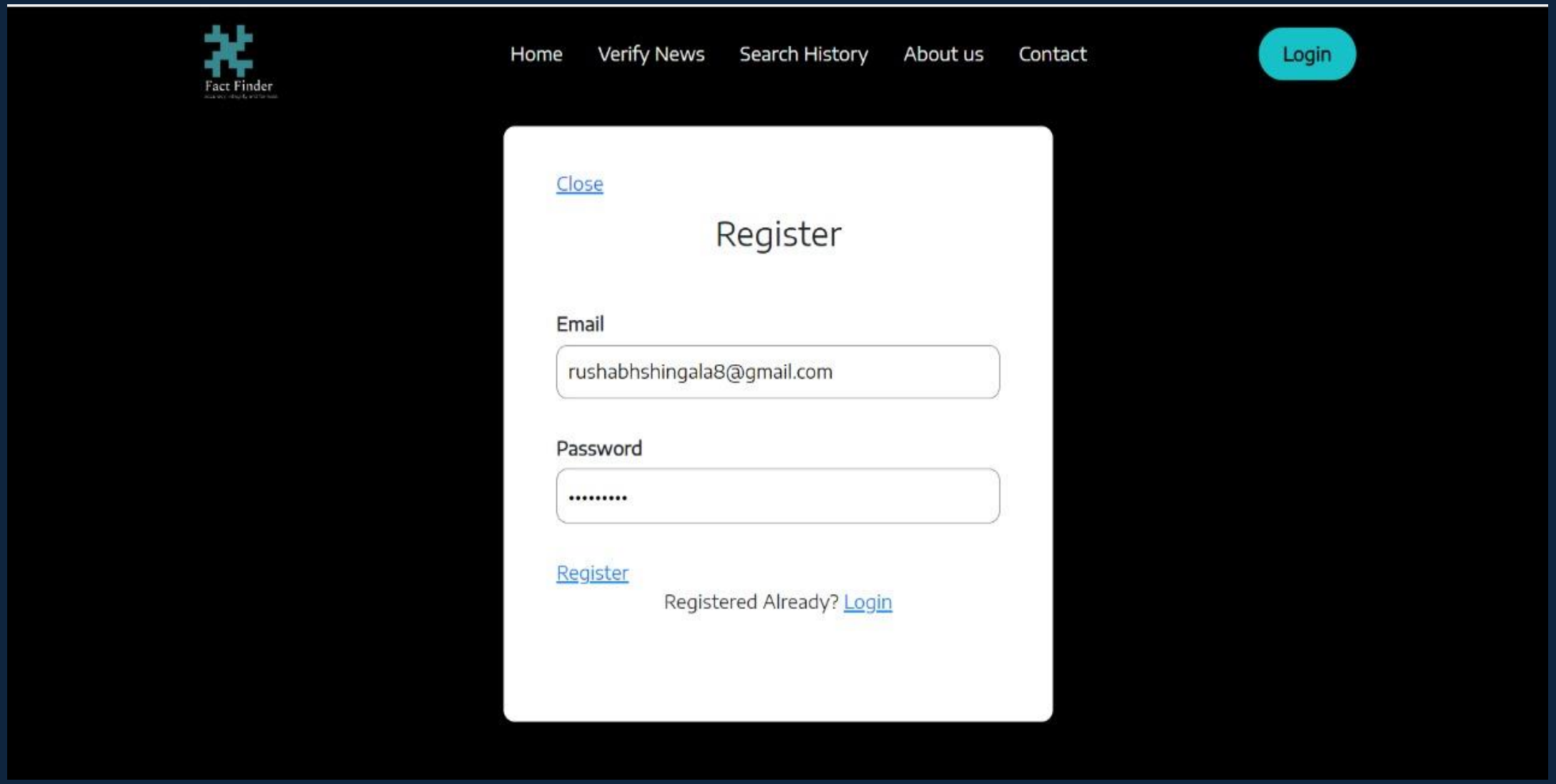
FactFinder website screenshots:



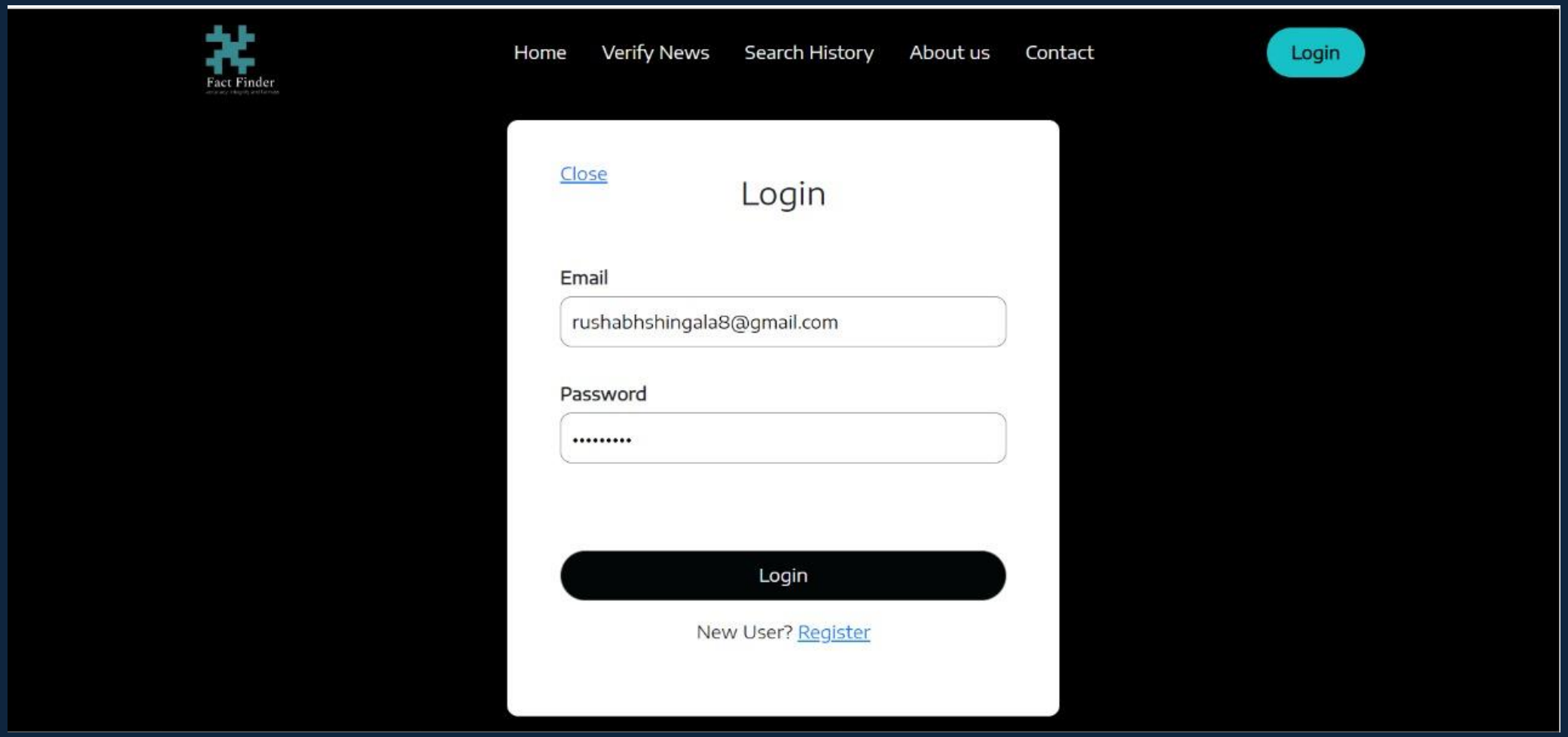
FactFinder website screenshots:



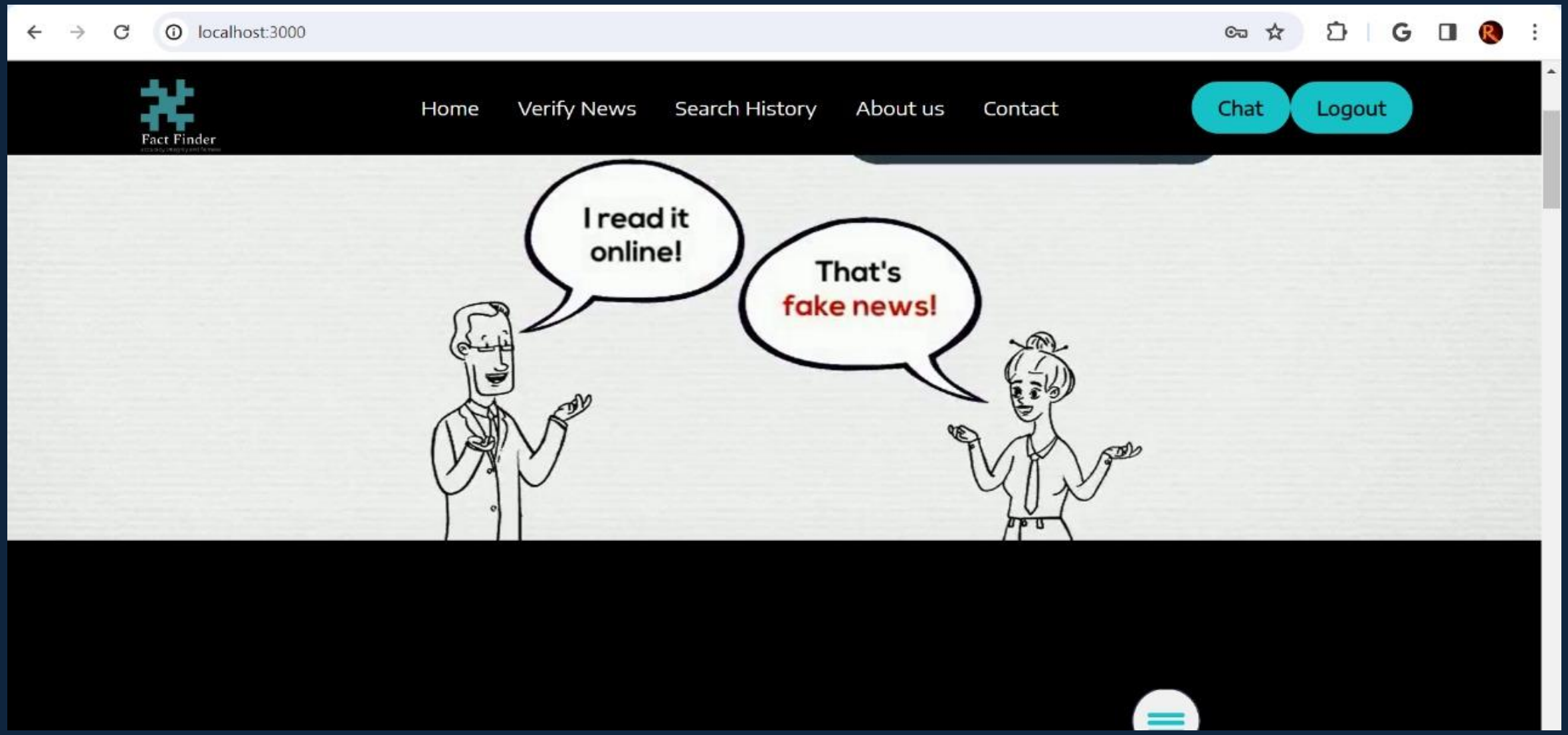
FactFinder website screenshots:



FactFinder website screenshots:



FactFinder website screenshots:



API

```python
app = Flask(__name__)



@app.route('/')
def index():
    return render_template('index.html')

@app.route('/process', methods=['POST'])
def process():
    if request.method == 'POST':
        data = request.json['data']


        result = prediction(data)
        return result


def prediction(data):
    news_data = {'news_to_predict': [data]}
    df_news = pd.DataFrame(news_data)
    df_news['news_to_predict'] = df_news['news_to_predict'].apply(clean_and_lower)
    news_X_test = df_news['news_to_predict']
    news_X_test = vectorization.transform(news_X_test)
    result = model.predict(news_X_test)



    if result[0] == 1:

        return "This News is Fake"
    else:
        return "This is an Actual News"
```

API (Cont..)

```python
def prediction(data):
    news_data = {'news_to_predict': [data]}
    df_news = pd.DataFrame(news_data)
    df_news['news_to_predict'] = df_news['news_to_predict'].apply(clean_and_lower)
    news_X_test = df_news['news_to_predict']
    news_X_test = vectorization.transform(news_X_test)
    result = model.predict(news_X_test)



    if result[0] == 1:

        return "This News is Fake"
    else:
        return "This is an Actual News"

if __name__ == '__main__':
    app.run(debug=True)

print("all done main.py")
```

Git Hub link:

https://github.com/htmw/2024S-JusticeLeague/wiki

Live Application Demo

Sprint 3 Demo (youtube.com)

https://www.youtube.com/watch?v=HYKTxQoSWUI

Fact Finder
accuracy, integrity and fairness




Thank you

Team Justice League