Project Title

Project Documentation

1. Introduction

• Project title : Citizen AI - Intelligent Citizen Engagement Platform
• Team member : Heerthini - S
• Team member : Gnana Thirisha - T
• Team member : Indhumathi - R
• Team member : Bharathi - C

2. Project Overview

• Purpose :
Citizen AI uses IBM Granite models to provide quick, helpful answers about government services and civic issues. It also tracks public sentiment and presents dashboards for officials to visualize feedback. The project is designed to run on Google Colab for easy, low-cost setup and reliable performance.

• Features:
Conversational Assistance
Key Point: Quick response to civic queries
Functionality: Provides natural language answers about government services.

Sentiment Tracking
Key Point: Public opinion insights
Functionality: Collects and analyzes citizen sentiment for officials.

Dashboard Visualization
Key Point: Data transparency
Functionality: Displays feedback and insights in simple dashboards.

Lightweight Deployment
Key Point: Easy setup
Functionality: Runs seamlessly in Google Colab with GPU support.

3. Architecture

Frontend (Gradio):
Provides an interactive interface for citizens to ask questions and officials to view dashboards.

Backend (IBM Granite Models via Hugging Face):
Granite models are used for natural language understanding and generating responses.

Deployment (Google Colab):
Colab provides a low-cost, GPU-enabled environment to host and run the application.

Version Control (GitHub):
The project is versioned and stored in GitHub for collaboration and updates.

4. Setup Instructions

Prerequisites:

o Python 3.9 or later
o Gradio Framework
o IBM Granite Models access via Hugging Face
o Git installed
o Google Colab account with T4 GPU enabled

Installation Process:
o Open Google Colab and create a new notebook
o Change runtime to GPU (T4)
o Install dependencies: !pip install transformers torch gradio
o Run provided Citizen AI code cells
o Access the generated Gradio app link
o Upload project to GitHub repository

5. Folder Structure

app/ – Gradio application scripts
notebooks/ – Google Colab notebooks
models/ – Hugging Face Granite model references
.github/ – GitHub configuration files
citizen_ai.py – Main application file
requirements.txt – Dependencies list

6. Running the Application

➢ Open the Google Colab notebook
➢ Install required dependencies
➢ Run the notebook cells sequentially
➢ Launch the Gradio app from the output link
➢ View dashboards and interact with the chatbot
➢ Push project files to GitHub repository

7. API Documentation

Citizen AI does not use custom APIs but relies on Hugging Face Granite model APIs and Gradio interface.

8. Authentication

Basic setup runs in open mode for demo purposes.
Future enhancements can integrate authentication using:
• API keys
• OAuth2 with IBM Cloud credentials
• Role-based access for citizens and officials

9. User Interface

The Gradio-based interface is simple and accessible:
• Chatbot interface for citizens
• Dashboard for officials to view sentiment and feedback
• Real-time updates through Google Colab hosting

10. Testing

Testing includes:
• Unit Testing: Verifying code in Colab cells
• Manual Testing: Checking chatbot responses and dashboard outputs
• Deployment Testing: Running app on Colab with GPU

11. Screenshots

(Screenshots can be inserted from Colab and Gradio outputs)

12. Known Issues

• Limited to Colab runtime sessions (temporary environment)
• Requires stable internet connection
• Dependent on Hugging Face model availability

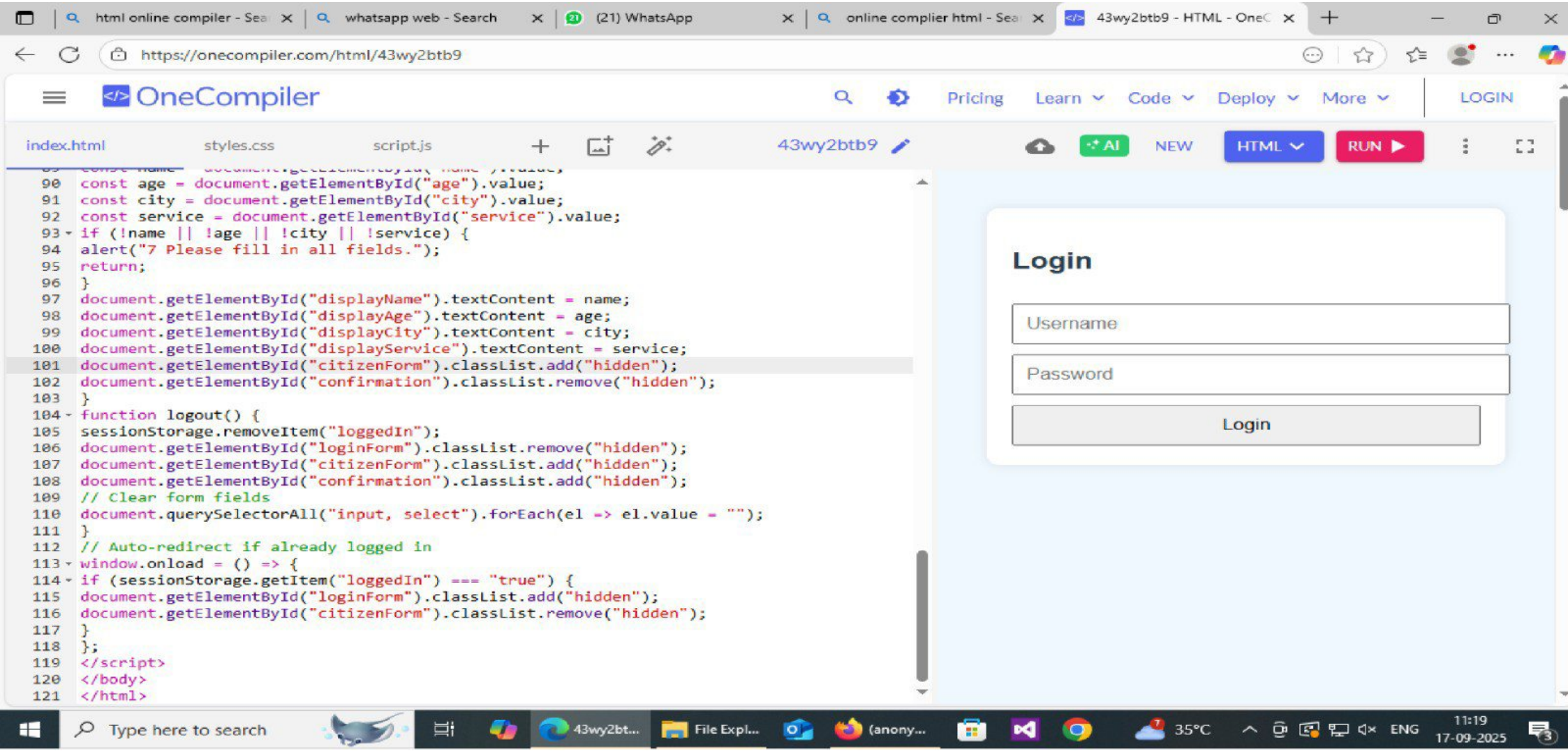
13. Future Enhancements

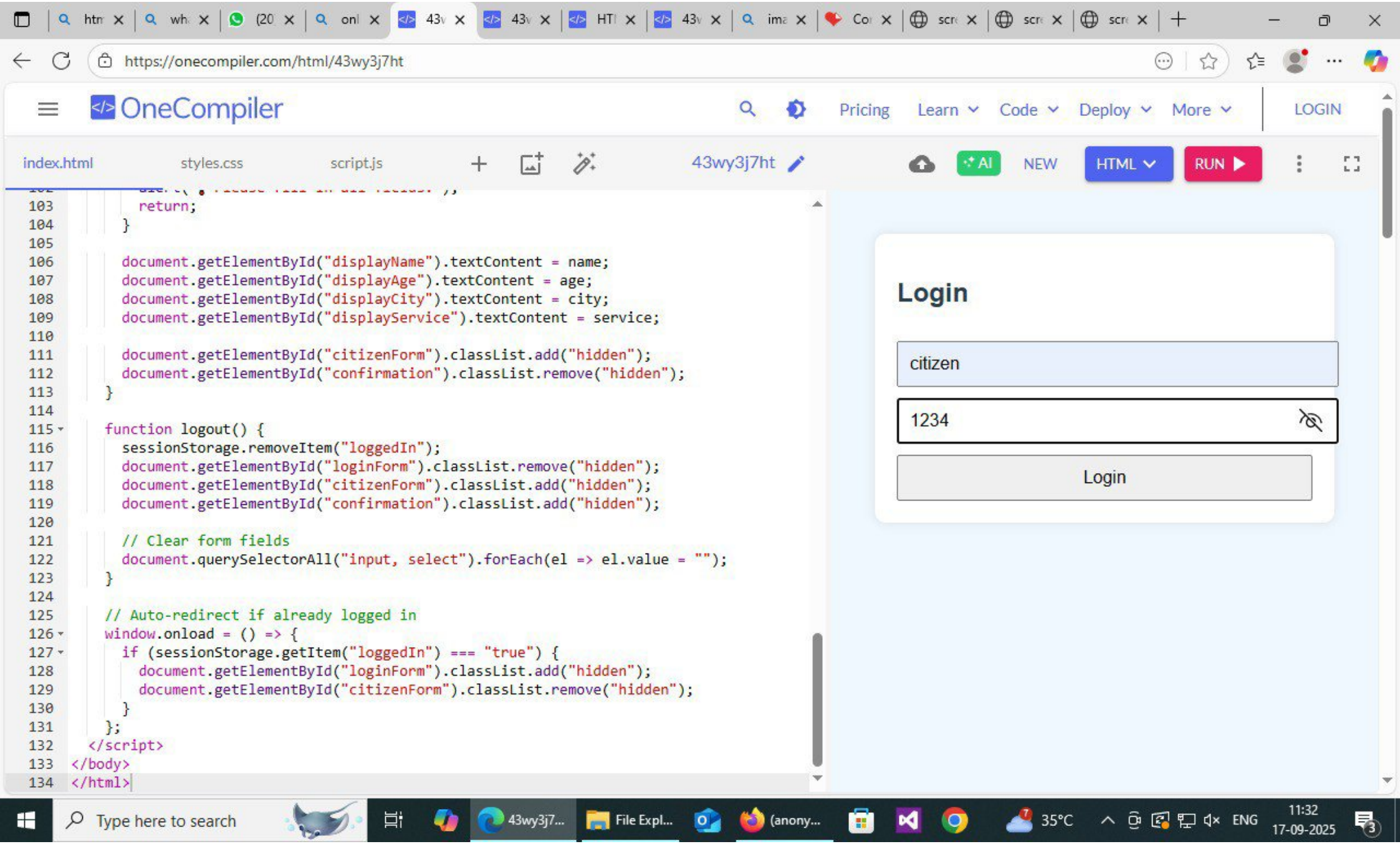
• Dedicated cloud deployment (IBM Cloud, AWS, or Azure)
• Advanced dashboards with analytics
• Multi-language citizen support
• Integration with real government data sources

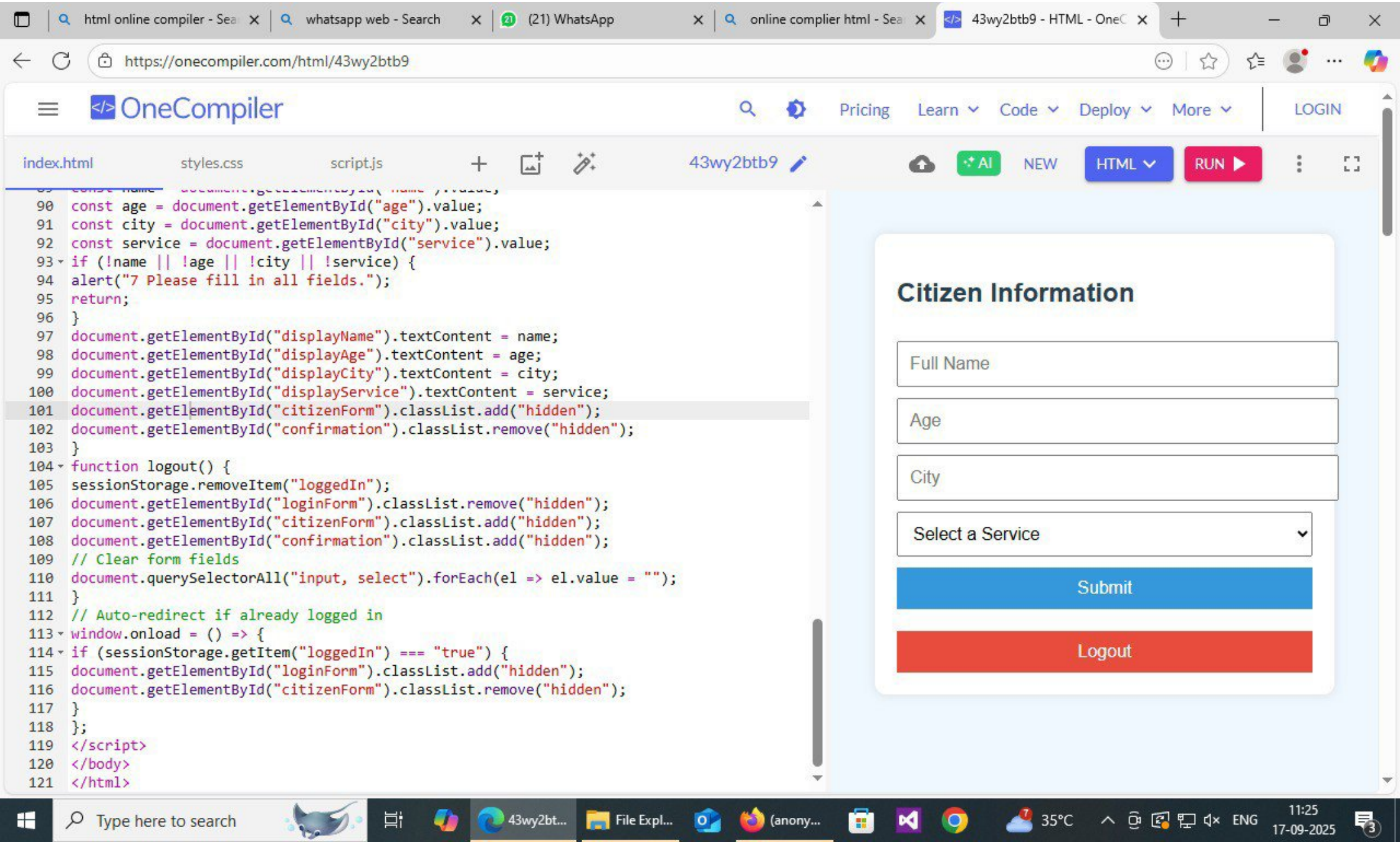
```
                plan_output = gr.Textbox(label="Personalized Treatment Plan", lines=20)

        plan_btn.click(treatment_plan, inputs=[condition_input, age_input, gender_input, history_input], outputs=plan_output)

app.launch(share=True)
```

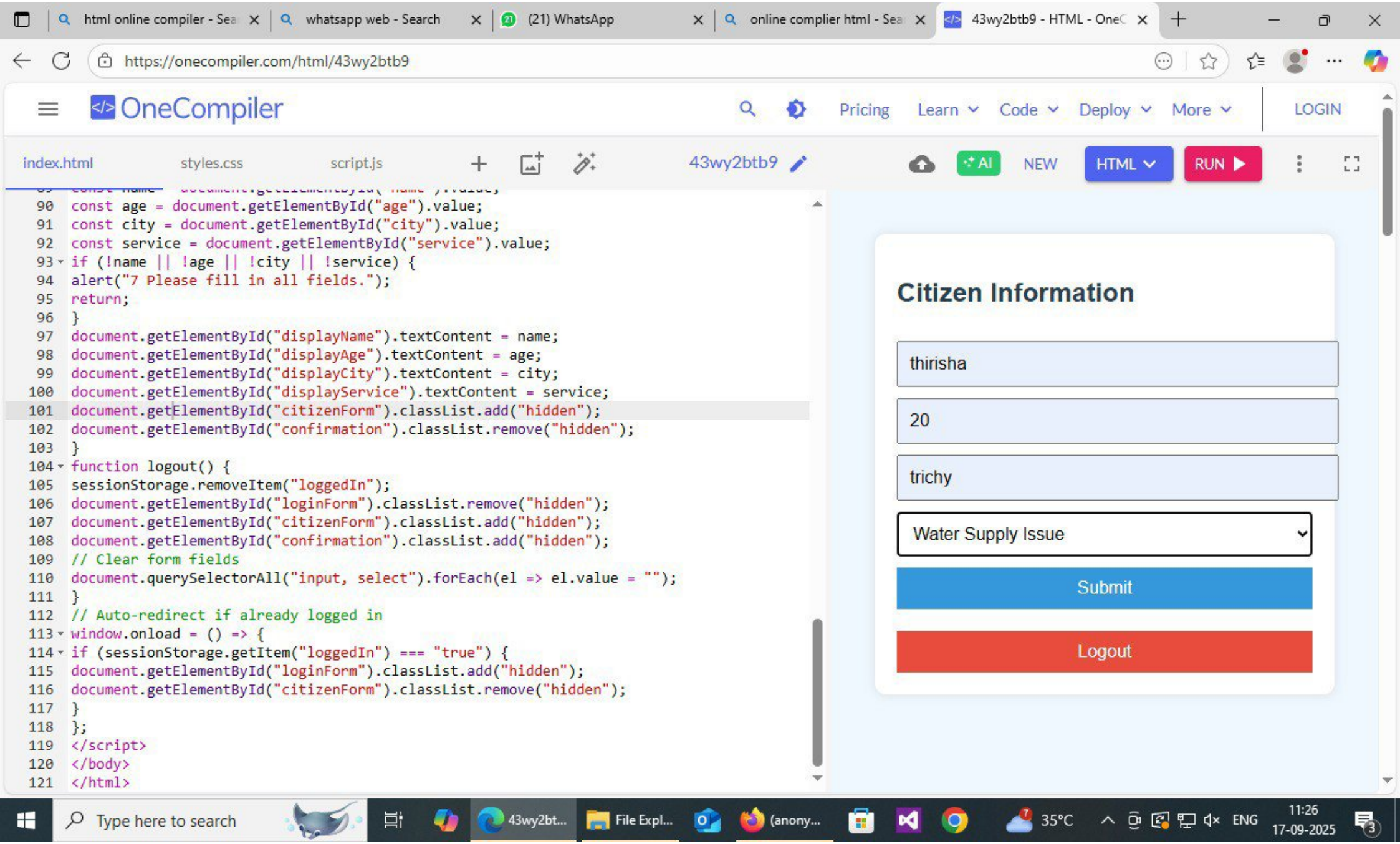
```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Cola
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
`torch_dtype` is deprecated! Use `dtype` instead!
Loading checkpoint shards: 100% 2/2 [00:57<00:00, 23.83s/it]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://38bc3259d0356c8cb8.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working directory to deploy to H
```

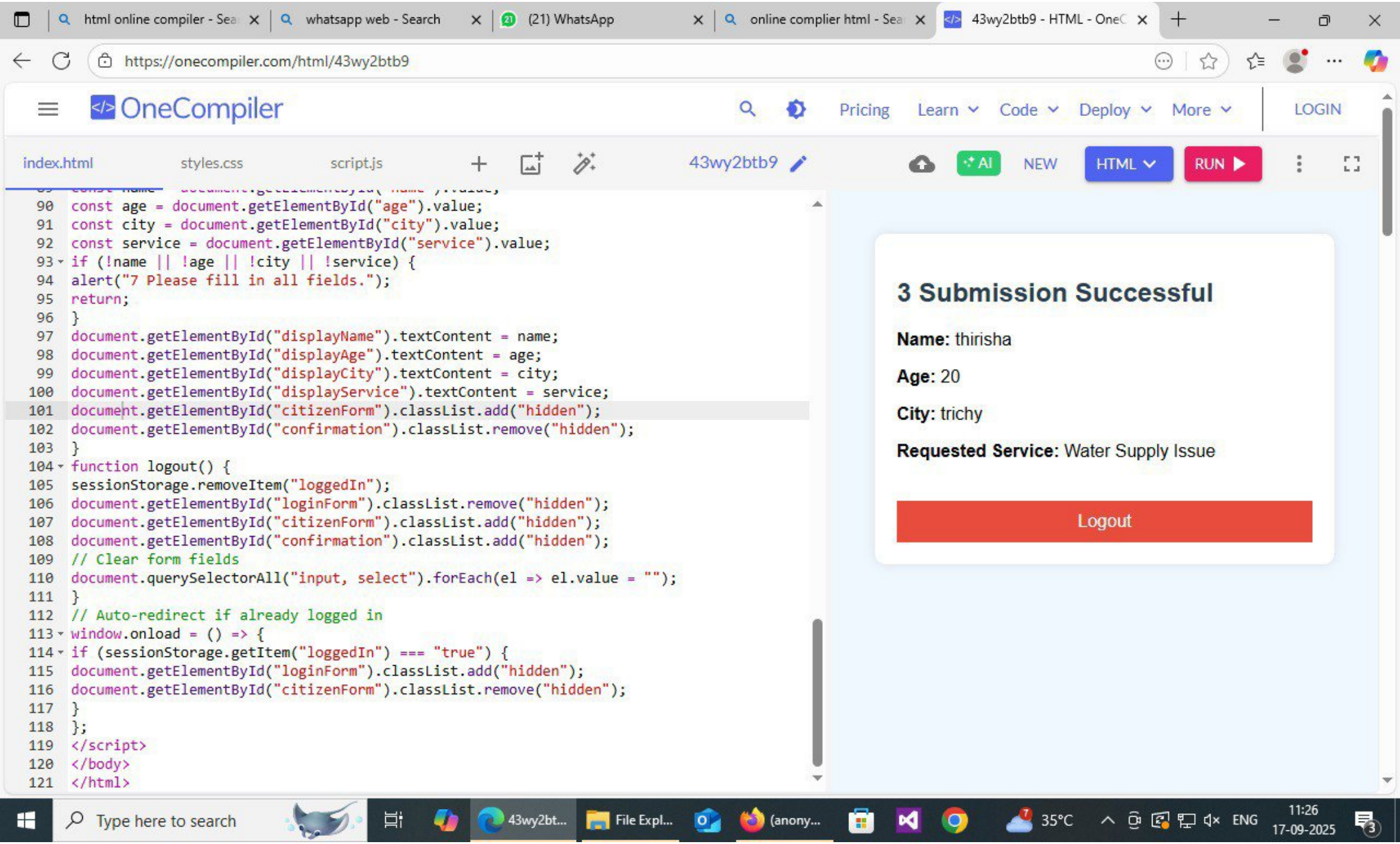
e.g., fever, headache, cough, fatigue...


html online compiler - Sea
whatsapp web - Search
(21) WhatsApp
online complier html - Sea
43wy2btb9 - HTML - OneC
https://onecompiler.com/html/43wy2btb9
OneCompiler
Pricing
Learn
Code
Deploy
More
LOGIN
index.html
styles.css
script.js
43wy2btb9
AI
NEW
HTML
RUN
const age = document.getElementById("age").value;
const city = document.getElementById("city").value;
const service = document.getElementById("service").value;
if (!name || !age || !city || !service) {
alert("7 Please fill in all fields.");
return;
}
document.getElementById("displayName").textContent = name;
document.getElementById("displayAge").textContent = age;
document.getElementById("displayCity").textContent = city;
document.getElementById("displayService").textContent = service;
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.remove("hidden");
}
function logout() {
sessionStorage.removeItem("loggedIn");
document.getElementById("loginForm").classList.remove("hidden");
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.add("hidden");
// Clear form fields
document.querySelectorAll("input, select").forEach(el => el.value = "");
}
// Auto-redirect if already logged in
window.onload = () => {
if (sessionStorage.getItem("loggedIn") === "true") {
document.getElementById("loginForm").classList.add("hidden");
document.getElementById("citizenForm").classList.remove("hidden");
}
};
</script>
</body>
</html>
Login
Username
Password
Login
Type here to search
43wy2bt...
File Expl...
(anony...
35°C
ENG
11:19
17-09-2025


https://onecompiler.com/html/43wy3j7ht
OneCompiler
Pricing
Learn
Code
Deploy
More
LOGIN
index.html
styles.css
script.js
43wy3j7ht
AI
NEW
HTML
RUN
return;
}
document.getElementById("displayName").textContent = name;
document.getElementById("displayAge").textContent = age;
document.getElementById("displayCity").textContent = city;
document.getElementById("displayService").textContent = service;
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.remove("hidden");
}
function logout() {
sessionStorage.removeItem("loggedIn");
document.getElementById("loginForm").classList.remove("hidden");
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.add("hidden");
// Clear form fields
document.querySelectorAll("input, select").forEach(el => el.value = "");
}
// Auto-redirect if already logged in
window.onload = () => {
if (sessionStorage.getItem("loggedIn") === "true") {
document.getElementById("loginForm").classList.add("hidden");
document.getElementById("citizenForm").classList.remove("hidden");
}
};
</script>
</body>
</html>
Login
citizen
1234
Login
Type here to search
35°C
ENG
11:32
17-09-2025


html online compiler - Sea
whatsapp web - Search
(21) WhatsApp
online complier html - Sea
43wy2btb9 - HTML - OneC
https://onecompiler.com/html/43wy2btb9
OneCompiler
Pricing
Learn
Code
Deploy
More
LOGIN
index.html
styles.css
script.js
43wy2btb9
AI
NEW
HTML
RUN
const age = document.getElementById("age").value;
const city = document.getElementById("city").value;
const service = document.getElementById("service").value;
if (!name || !age || !city || !service) {
alert("7 Please fill in all fields.");
return;
}
document.getElementById("displayName").textContent = name;
document.getElementById("displayAge").textContent = age;
document.getElementById("displayCity").textContent = city;
document.getElementById("displayService").textContent = service;
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.remove("hidden");
}
function logout() {
sessionStorage.removeItem("loggedIn");
document.getElementById("loginForm").classList.remove("hidden");
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.add("hidden");
// Clear form fields
document.querySelectorAll("input, select").forEach(el => el.value = "");
}
// Auto-redirect if already logged in
window.onload = () => {
if (sessionStorage.getItem("loggedIn") === "true") {
document.getElementById("loginForm").classList.add("hidden");
document.getElementById("citizenForm").classList.remove("hidden");
}
};
</script>
</body>
</html>
Citizen Information
Full Name
Age
City
Select a Service
Submit
Logout
Type here to search
43wy2bt...
File Expl...
(anony...
35°C
ENG
11:25
17-09-2025


html online compiler - Sea
whatsapp web - Search
(21) WhatsApp
online complier html - Sea
43wy2btb9 - HTML - OneC
https://onecompiler.com/html/43wy2btb9
OneCompiler
Pricing
Learn
Code
Deploy
More
LOGIN
index.html
styles.css
script.js
43wy2btb9
AI
NEW
HTML
RUN
const age = document.getElementById("age").value;
const city = document.getElementById("city").value;
const service = document.getElementById("service").value;
if (!name || !age || !city || !service) {
alert("7 Please fill in all fields.");
return;
}
document.getElementById("displayName").textContent = name;
document.getElementById("displayAge").textContent = age;
document.getElementById("displayCity").textContent = city;
document.getElementById("displayService").textContent = service;
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.remove("hidden");
}
function logout() {
sessionStorage.removeItem("loggedIn");
document.getElementById("loginForm").classList.remove("hidden");
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.add("hidden");
// Clear form fields
document.querySelectorAll("input, select").forEach(el => el.value = "");
}
// Auto-redirect if already logged in
window.onload = () => {
if (sessionStorage.getItem("loggedIn") === "true") {
document.getElementById("loginForm").classList.add("hidden");
document.getElementById("citizenForm").classList.remove("hidden");
}
};
</script>
</body>
</html>
Citizen Information
thirisha
20
trichy
Water Supply Issue
Submit
Logout
Type here to search
43wy2bt...
File Expl...
(anony...
35°C
ENG
11:26
17-09-2025


html online compiler - Sea
whatsapp web - Search
(21) WhatsApp
online complier html - Sea
43wy2btb9 - HTML - OneC
https://onecompiler.com/html/43wy2btb9
OneCompiler
Pricing
Learn
Code
Deploy
More
LOGIN
index.html
styles.css
script.js
43wy2btb9
AI
NEW
HTML
RUN
const age = document.getElementById("age").value;
const city = document.getElementById("city").value;
const service = document.getElementById("service").value;
if (!name || !age || !city || !service) {
alert("7 Please fill in all fields.");
return;
}
document.getElementById("displayName").textContent = name;
document.getElementById("displayAge").textContent = age;
document.getElementById("displayCity").textContent = city;
document.getElementById("displayService").textContent = service;
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.remove("hidden");
}
function logout() {
sessionStorage.removeItem("loggedIn");
document.getElementById("loginForm").classList.remove("hidden");
document.getElementById("citizenForm").classList.add("hidden");
document.getElementById("confirmation").classList.add("hidden");
// Clear form fields
document.querySelectorAll("input, select").forEach(el => el.value = "");
}
// Auto-redirect if already logged in
window.onload = () => {
if (sessionStorage.getItem("loggedIn") === "true") {
document.getElementById("loginForm").classList.add("hidden");
document.getElementById("citizenForm").classList.remove("hidden");
}
};
</script>
</body>
</html>
3 Submission Successful
Name: thirisha
Age: 20
City: trichy
Requested Service: Water Supply Issue
Logout
Type here to search
43wy2bt...
File Expl...
(anony...
35°C
ENG
11:26
17-09-2025

Project Title

Project Documentation

1. Introduction

• Project title : Citizen AI - Intelligent Citizen Engagement Platform
• Team member : Heerthini - S
• Team member : Gnana Thirisha - T
• Team member : Indhumathi - R
• Team member : Bharathi - C

2. Project Overview

• Purpose :
Citizen AI uses IBM Granite models to provide quick, helpful answers about government services and civic issues. It also tracks public sentiment and presents dashboards for officials to visualize feedback. The project is designed to run on Google Colab for easy, low-cost setup and reliable performance.

• Features:
Conversational Assistance
Key Point: Quick response to civic queries
Functionality: Provides natural language answers about government services.

Sentiment Tracking
Key Point: Public opinion insights
Functionality: Collects and analyzes citizen sentiment for officials.

Dashboard Visualization
Key Point: Data transparency
Functionality: Displays feedback and insights in simple dashboards.

Lightweight Deployment
Key Point: Easy setup
Functionality: Runs seamlessly in Google Colab with GPU support.

3. Architecture

Frontend (Gradio):
Provides an interactive interface for citizens to ask questions and officials to view dashboards.

Backend (IBM Granite Models via Hugging Face):
Granite models are used for natural language understanding and generating responses.

Deployment (Google Colab):
Colab provides a low-cost, GPU-enabled environment to host and run the application.

Version Control (GitHub):
The project is versioned and stored in GitHub for collaboration and updates.

4. Setup Instructions

Prerequisites:

o Python 3.9 or later
o Gradio Framework
o IBM Granite Models access via Hugging Face
o Git installed
o Google Colab account with T4 GPU enabled

Installation Process:
o Open Google Colab and create a new notebook
o Change runtime to GPU (T4)
o Install dependencies: !pip install transformers torch gradio
o Run provided Citizen AI code cells
o Access the generated Gradio app link
o Upload project to GitHub repository

5. Folder Structure

app/ – Gradio application scripts
notebooks/ – Google Colab notebooks
models/ – Hugging Face Granite model references
.github/ – GitHub configuration files
citizen_ai.py – Main application file
requirements.txt – Dependencies list

6. Running the Application

➢ Open the Google Colab notebook
➢ Install required dependencies
➢ Run the notebook cells sequentially
➢ Launch the Gradio app from the output link
➢ View dashboards and interact with the chatbot
➢ Push project files to GitHub repository

7. API Documentation

Citizen AI does not use custom APIs but relies on Hugging Face Granite model APIs and Gradio interface.

8. Authentication

Basic setup runs in open mode for demo purposes.
Future enhancements can integrate authentication using:
• API keys
• OAuth2 with IBM Cloud credentials
• Role-based access for citizens and officials

9. User Interface

The Gradio-based interface is simple and accessible:
• Chatbot interface for citizens
• Dashboard for officials to view sentiment and feedback
• Real-time updates through Google Colab hosting

10. Testing

Testing includes:
• Unit Testing: Verifying code in Colab cells
• Manual Testing: Checking chatbot responses and dashboard outputs
• Deployment Testing: Running app on Colab with GPU

11. Screenshots

(Screenshots can be inserted from Colab and Gradio outputs)

12. Known Issues

• Limited to Colab runtime sessions (temporary environment)
• Requires stable internet connection
• Dependent on Hugging Face model availability

13. Future Enhancements

• Dedicated cloud deployment (IBM Cloud, AWS, or Azure)
• Advanced dashboards with analytics
• Multi-language citizen support
• Integration with real government data sources