

CITIZEN AI – INTELLIGENT CITIZEN ENGAGEMENT PLATFORM

1.Introduction :

- Project title : Citizen AI-Intelligent Citizen Engagement Platform
- Team member: SARASWATHI.S
- Team member : JANANI.E
- Team member : DHANALAKSHMI.S
- Team member : KANISHKA.M

2.project overview :

The Citizen AI-Intelligent Citizen Engagement Platform is designed to improve communication between citizens and government services through artificial intelligence. Its primary purpose is to provide a seamless and efficient way for citizens to interact with government agencies, access public services, and report concerns.

1. Real-Time AI Chatbot: allows citizens to ask questions and receive immediate responses about public services, documents, and civic procedures
2. Citizen Sentiment Analysis: analyzes public sentiment from submitted feedback to understand citizen satisfaction and concerns
3. -Interactive Dashboard: visualizes feedback trends and sentiment data to help government agencies make informed decisions
4. Context-Aware Responses: provides accurate and relevant responses based on the conversation flow
5. Concern Management: enables citizens to report issues and concerns, which are then tracked and prioritized by government agencies
6. Admin Dashboard: provides a centralized platform for government agencies to manage citizen interactions, sentiment analysis, and concern tracking

Benefits:

1. Improved Citizen Engagement: fosters a more interactive and responsive relationship between citizens and government agencies
2. Enhanced Public Services: streamlines access to public services and information, making it easier for citizens to navigate government bureaucracy
3. Data-Driven Decision Making: provides government agencies with valuable insights into citizen sentiment and concerns, enabling data-driven decision making
4. Increased Transparency: promotes transparency and accountability in government agencies through interactive dashboards and sentiment analysis

Overall, the Citizen AI platform aims to create a more citizen-centric and responsive government by leveraging AI-powered technologies to improve communication, engagement, and public services.

3. Architecture :

The architecture of the Citizen AI-Intelligent Citizen Engagement Platform is designed to facilitate seamless interaction between citizens and government agencies. Here's an overview of the platform's architecture:

Components:

1. User Interface (UI)
2. AI-Powered Chatbot
3. Natural Language Processing (NLP)
4. Sentiment Analysis
5. Data Analytics
6. Integration Layer
7. Security and Authentication

Architecture:

1. Frontend: The UI is built using modern web technologies, providing a responsive and intuitive experience for citizens.
2. Backend: The backend is built using a robust framework, handling requests, processing data, and integrating with external systems.
3. AI Engine: The AI engine powers the chatbot, sentiment analysis, and data analytics, enabling intelligent and personalized interactions.
4. Data Storage: A secure and scalable data storage solution stores citizen interactions, feedback, and sentiment data.

4. Set-up Instructions :

Prerequisites:

1. Server Requirements: Ensure you have a server with sufficient resources (CPU, RAM, and storage) to support the platform's requirements.
2. Database: Set up a database management system (e.g., MySQL, PostgreSQL) to store citizen data, interactions, and feedback.
3. Cloud Infrastructure*: Ensure you have a cloud infrastructure (e.g., AWS, Azure, Google Cloud) to host the platform and scale as needed.
4. Security: Implement robust security measures, including encryption, firewalls, and access controls, to protect citizen data and prevent unauthorized access.
5. AI and ML Frameworks: Install necessary AI and ML frameworks (e.g., TensorFlow, PyTorch) to support the platform's AI-powered features.

Installation Process:

1. Clone the Repository: Clone the Citizen AI repository from a version control system (e.g., GitHub).
2. Configure Environment Variables: Configure environment variables, such as database connections, API keys, and cloud infrastructure settings.

3. Install Dependencies: Install required dependencies, including libraries and frameworks, using a package manager (e.g., pip, npm).
4. Run Migrations: Run database migrations to set up the schema and populate initial data.
5. Deploy to Cloud: Deploy the platform to your cloud infrastructure, configuring load balancers, auto-scaling, and monitoring as needed.
6. Configure AI Models: Configure and train AI models, such as chatbots and sentiment analysis, using provided datasets or your own data.
7. Test and Validate: Test and validate the platform's functionality, performance, and security to ensure it meets requirements.

5. Folder Structure :

Root Directory

- `citizen-ai/`

Subdirectories

1. Backend

- `backend/`

- `app/` (application code)

- `config/` (configuration files)

- `models/` (database models)

- `routes/` (API routes)

- `services/` (business logic)

- `utils/` (utility functions)

2. Frontend

- `frontend/`
 - `public/` (static assets)
 - `src/` (source code)
 - `components/` (UI components)
 - `containers/` (container components)
 - `actions/` (action creators)
 - `reducers/` (reducers)
 - `utils/` (utility functions)

3. AI and ML

- `ai-ml/`
 - `models/` (AI and ML models)
 - `training/` (model training scripts)
 - `evaluation/` (model evaluation scripts)

4. Database

- `database/`
 - `schema/` (database schema)
 - `migrations/` (database migrations)

5. Documentation

- `docs/`
 - `api/` (API documentation)
 - `user/` (user documentation)
 - `developer/` (developer documentation)

6. Running the Application:

Running the Backend

1. Navigate to the backend directory: ``cd backend``
2. Run the backend server: ``python app.py`` (or the command specified in your backend framework)

Running the Frontend

1. Navigate to the frontend directory: ``cd frontend``
2. Run the frontend development server: ``npm start`` (or the command specified in your frontend framework)

Running the AI and ML Components

1. Navigate to the AI and ML directory: ``cd ai-ml``
2. Run the AI and ML models: ``python train.py`` (or the command specified in your AI and ML framework)

By following these steps, you can run the Citizen AI application and test its functionality. Ensure you have the necessary dependencies installed and configured environment variables before running the application.

7.API Documentation :

- POST /chat/ask: Submit a question to the AI chatbot and receive a response
- GET /chat/history: Retrieve chat history
- POST /feedback/submit: Submit feedback on government services
- GET /feedback/analyze: Analyze sentiment of submitted feedback
- POST /concern/submit: Report a concern or issue
- GET /concern/list: List all reported concerns
- GET /concern/{id}: Retrieve a specific concern by ID

8. Authentication:

- GET /auth/login: Login page for administrators
- POST /auth/login: Process login credentials
- GET /auth/logout: Logout

9. User Interface:

The user interface (UI) of the Citizen AI-Intelligent Citizen Engagement Platform is designed to provide a seamless and intuitive experience for citizens to interact with government services. Here are some key features of the UI

1. Chatbot Interface
2. Feedback Mechanism:
3. Concern Reporting
4. Personalized Dashboard
5. User-Friendly Navigation
6. Accessibility Features

Design Principles:

1. Citizen-Centric
2. Simple and Intuitive
3. Consistent
4. Accessible

Benefits:

1. Improved Citizen Engagement
2. Increased Transparency
3. Enhanced User Experience

10. Testing :

1. Unit Testing: Verify individual components or units of code function correctly.
2. Integration Testing: Test interactions between components or systems.
3. Functional Testing: Validate platform functionality meets requirements.
4. Performance Testing: Evaluate platform performance under various loads.
5. Security Testing: Identify vulnerabilities and ensure data protection.

Testing Tools

1. Automated Testing Frameworks: Use frameworks like Selenium or Cypress for automated testing.
2. Manual Testing: Perform manual testing to identify issues not caught by automated tests.

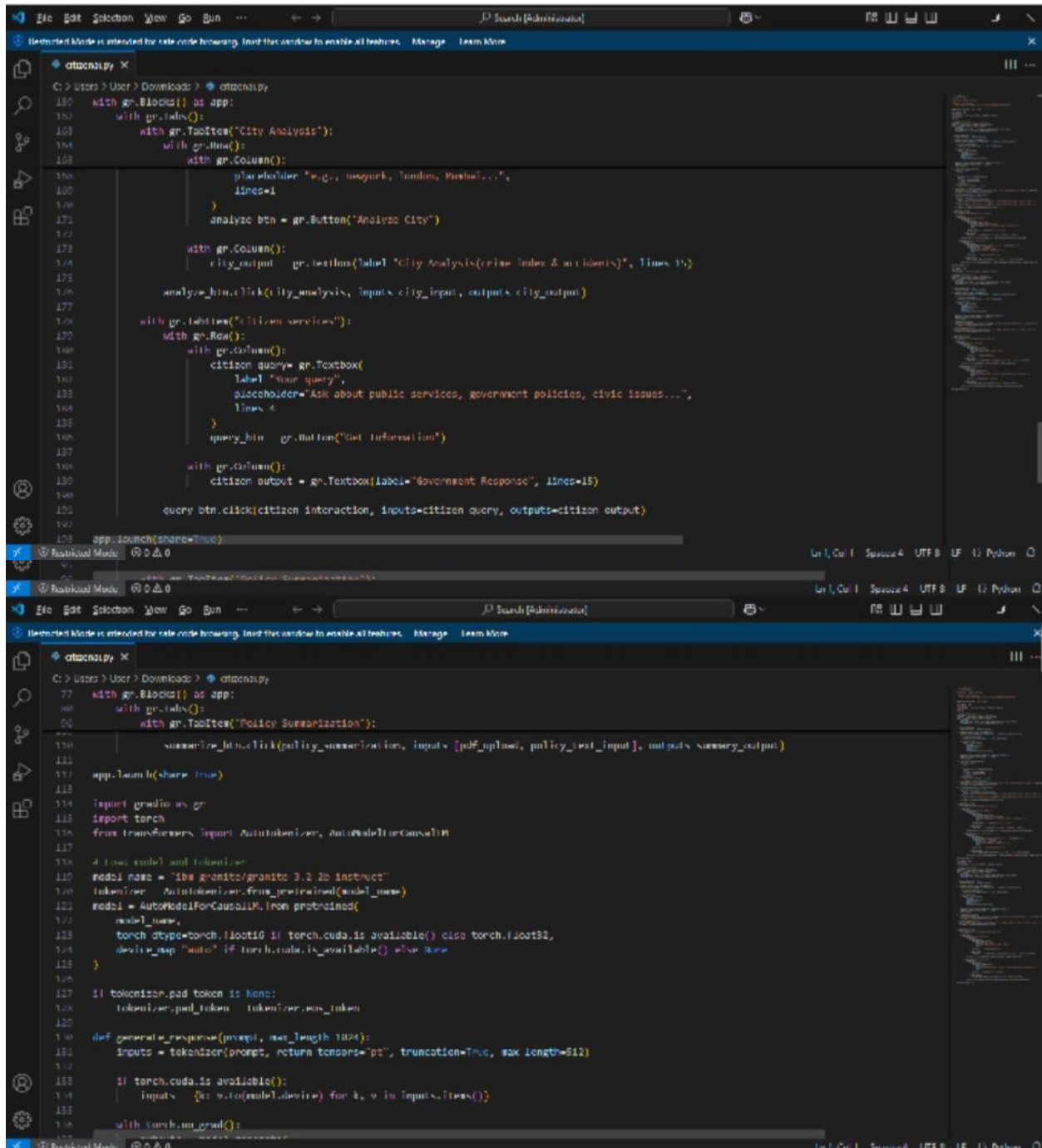
11. Known Issues :

- Security and Data Protection Issues
- AI Model – Related Issues
- Operational and Technical Issues
- User Experience Issues

12. Future Enhancement:

- Advanced AI – Powered Chatbots
- Prediction Analytics
- Personalized Citizen Experiences
- Multilingual Support
- Sentiment Analysis
- Integration with Emerging Technologies
- Enhanced Security Measures

13. Screenshot:



```

C:\Users\User> Download > otocnslpy
150 with gr.Blocks() as app:
151     with gr.Tab():
152         with gr.TabItem("City Analysis"):
153             with gr.Column():
154                 with gr.Column():
155                     placeholder = gr.Textbox(placeholder="e.g., newyork, london, paris...", lines=1)
156                 )
157                 analyze_btn = gr.Button("Analyze City")
158             )
159             with gr.Column():
160                 city_output = gr.Textbox(label="City Analysis (crime index & accidents)", lines=10)
161             )
162             analyze_btn.click(city_analysis, inputs=[city_input], outputs=[city_output])
163         )
164     with gr.TabItem("Public Services"):
165         with gr.Row():
166             with gr.Column():
167                 citizen_query = gr.Textbox(
168                     label="Your query",
169                     placeholder="Ask about public services, government policies, civic issues...",
170                     lines=4
171                 )
172                 query_btn = gr.Button("Get Information")
173             )
174             with gr.Column():
175                 citizen_output = gr.Textbox(label="Government Response", lines=15)
176             )
177             query_btn.click(citizen_interaction, inputs=[citizen_query], outputs=[citizen_output])
178         )
179 app.launch(share=True)
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```