

## GenAI Curriculum Generator

### AI-Powered Educational Curriculum Design with IBM Granite 3.3 2B

#### Project Description

**GenAI Curriculum Generator** is an intelligent curriculum design platform that leverages IBM's Granite 3.3 2B AI model (via Ollama) to provide comprehensive educational curriculum generation and planning recommendations. The platform addresses the challenge of accurate curriculum design by delivering AI-powered insights, detailed course structures, topic recommendations, and semester-wise syllabi.

Using Granite 3.3 2B's advanced language capabilities, the system analyzes educational parameters (skill, education level, number of semesters, weekly hours, industry focus), generates accurate course names, learning topics, course descriptions, and detailed curricula tailored to individual educational specifications. The system ensures fast response times through local Ollama deployment while maintaining high-quality curriculum advice through sophisticated AI analysis and educational algorithms.

**GenAI Curriculum Generator** transforms curriculum design into an intelligent, user-friendly experience through its modern interface, comprehensive feature set, and AI-powered analysis that provides personalized educational guidance considering skill complexity, learning level progression, and industry relevance.

## Scenarios

### **Scenario 1 – Machine Learning Masters Program:**

A university designs a 2-year Master's program in Machine Learning using the GenAI Curriculum Generator by entering parameters such as ML skill set, 4 semesters, 20–25 weekly hours, and AI industry focus. The Granite 3.3 2B model creates a logical progression of courses from basic machine learning to advanced topics like NLP and Reinforcement Learning. Each module contains structured topics such as CNNs, backpropagation, and optimization techniques. The system assigns 4 credits per course, builds semester-wise schedules, and defines capstone guidelines. Finally, a professional, publication-ready PDF syllabus is generated for admissions and student onboarding.

### **Scenario 2 – Full Stack Web Development Certification:**

A bootcamp develops a 6-month Full Stack Web Development certification by specifying certification level, 2 semesters, 30 weekly hours, and web industry focus. The Granite 3.3 2B model produces a beginner-to-advanced curriculum covering HTML, CSS, JavaScript, React, Node.js, and SQL. Each course includes practical, industry-oriented topics such as REST APIs, state management, and deployment strategies. The system organizes the learning path into a concise semester-wise schedule with project-based outcomes. An exportable curriculum document is delivered for immediate marketing.

### **Scenario 3 – Python for Data Science Program:**

An online platform creates a 1-year Python for Data Science program by providing inputs like Python skill, BTech level, 2 semesters, and data science industry focus. The AI model generates a balanced curriculum progressing from Python fundamentals to advanced data science applications. Courses include NumPy, Pandas, statistics, web scraping, and visualization with Matplotlib and Seaborn. Practical projects reinforce real-world data science skills. A professional PDF syllabus is produced to support student recruitment and promotions.

### **Scenario 4 – Artificial Intelligence Diploma Program:**

A technical institute launches a 3-year full-time AI diploma by entering AI skill, diploma level, 6 semesters, and AI R&D focus. The Granite 3.3 2B model generates an 18-course roadmap progressing from mathematical foundations to Machine Learning, Deep Learning, NLP, Computer Vision, and Reinforcement Learning. Each semester contains well-paced courses with 5–7 relevant topics. Research-oriented learning outcomes and capstone preparation are included. The institute receives a complete, accreditation-ready syllabus for official approval.

## Architecture Overview

**GenAI Curriculum Generator** is built as a modular platform combining Flask backend with IBM Granite 3.3 2B AI model for intelligent curriculum design. The architecture prioritizes accuracy, speed, and user experience by leveraging local AI inference and sophisticated curriculum generation algorithms.

## Core Technologies

- **Flask:** Lightweight Python web framework for routing and request processing
- **IBM Granite 3.3 2B:** Local AI inference via Ollama for intelligent curriculum generation
- **Ollama:** Local LLM runtime for Granite model execution
- **Python Algorithms:** Curriculum structure and validation logic
- **HTML5/CSS3/JavaScript:** Modern, responsive frontend with smooth animations
- **ReportLab:** Professional PDF document generation
- **Local Deployment:** No cloud API keys required, completely self-contained

## Pre-requisites

### Software Requirements

- **Python 3.8+:** Download from [python.org](https://python.org)
- **Ollama:** Download from [ollama.ai](https://ollama.ai)
- **Granite 3.3 2B Model:** Pre-downloaded via Ollama (ollama pull granite:3.3-2b)
- **Git:** Download from [git-scm.com](https://git-scm.com)
- **Code Editor:** VS Code, PyCharm, or any preferred IDE

### Knowledge Prerequisites

- **Python Basics:** Functions, classes, OOP, exception handling
- **Flask Framework:** Routing, request handling, JSON responses
- **HTML/CSS:** Responsive design, Flexbox/Grid layouts
- **JavaScript:** DOM manipulation, Fetch API, async/await
- **Educational Domain:** Basic understanding of curriculum design, learning outcomes
- **Mathematics:** Basic calculations, cost analysis

### Hardware Requirements

- **Processor:** Intel i5/AMD Ryzen 5 or better
- **RAM:** Minimum 8GB (16GB recommended for Ollama)
- **Storage:** 10GB free space (for Granite 3.3 2B model)
- **Internet:** Required for initial setup only

## Project Workflow

### Phase 1: Environment Setup & Ollama Configuration

Establish local AI infrastructure and validate model connectivity

#### Activity 1.1: Set up Python Environment

- Create project directory structure
- Initialize virtual environment
- Create requirements.txt with dependencies (Flask, requests, reportlab)
- Install all Python packages

#### Activity 1.2: Configure Ollama & Granite Model

- Download and install Ollama
- Pull Granite 3.3 2B model: [ollama pull granite:3.3-2b](#)
- Verify model accessibility on port 11434
- Test local inference capabilities with sample prompts

#### Activity 1.3: Validate System Configuration

- Test Flask application startup
- Verify Ollama API connectivity
- Test model response times (target: < 60 seconds)
- Document configuration steps for reference

## Phase 2: Core Backend Development

Build Flask infrastructure and curriculum generation logic

#### Activity 2.1: Set up Flask Application Structure

- Initialize Flask app with proper architecture
- Create route structure (/api/generate-curriculum, /api/download-pdf, /health)
- Implement CORS headers for frontend communication
- Set up request/response handling with JSON

#### Activity 2.2: Implement Ollama/Granite API Integration

- Create Ollama API communication module with error handling
- Implement prompt engineering for curriculum generation
- Add JSON response parsing and validation
- Handle API timeouts and implement fallback system

#### Activity 2.3: Build Curriculum Generation Engine

- Create single-batch curriculum generator function
- Implement course name generation logic using AI
- Build topic extraction and formatting system
- Develop curriculum structure builder (semesters, courses, credits)

### **Activity 2.4: Develop PDF Export Engine**

- Create ReportLab PDF generation system
- Implement professional table formatting
- Add semester-wise curriculum layout
- Build topic wrapping and text formatting

### **Activity 2.5: Build JSON Export Functionality**

- Create JSON serialization system
- Implement data validation
- Add file download handling
- Create backup data structure

## **Phase 3: Frontend Development**

**Create modern, responsive user interface**

### **Activity 3.1: Design Responsive HTML Structure**

- Create semantic HTML5 layout
- Design form sections for curriculum input
- Build results display templates
- Implement semester accordion structure

### **Activity 3.2: Develop CSS Styling & Animations**

- Implement modern gradient color scheme
- Create responsive grid layouts
- Add smooth transitions and animations
- Build mobile-responsive design
- Create topic tag styling

### **Activity 3.3: Create Interactive JavaScript Logic**

- Implement form validation
- Build real-time curriculum display
- Create semester accordion toggle
- Add PDF/JSON download functionality
- Implement loading and error states

### **Activity 3.4: Build Results Display Components**

- Create dynamic course cards
- Implement semester containers
- Build topic tag visualization
- Create summary card display
- Implement capstone project display

## Phase 4: Integration & Testing

**Integrate all components and comprehensive testing**

### Activity 4.1: End-to-End Integration

- Connect frontend to backend API
- Test API communication flow
- Validate data flow between components
- Test error handling and fallback system

### Activity 4.2: Functional Testing

- Test with various skills (ML, Web, Python, Data Science, Java)
- Validate different education levels (Diploma, BTech, Masters, Certification)
- Test all semester durations (2, 4, 6, 8 semesters)
- Verify PDF generation accuracy and formatting
- Test JSON export functionality

### Activity 4.3: Performance Testing

- Test Ollama response times (target: 15-30 seconds)
- Profile API call performance
- Optimize batch curriculum generation
- Load test with multiple concurrent requests

## Phase 5: Deployment & Documentation

**Deploy and document the system**

### Activity 5.1: Local Deployment

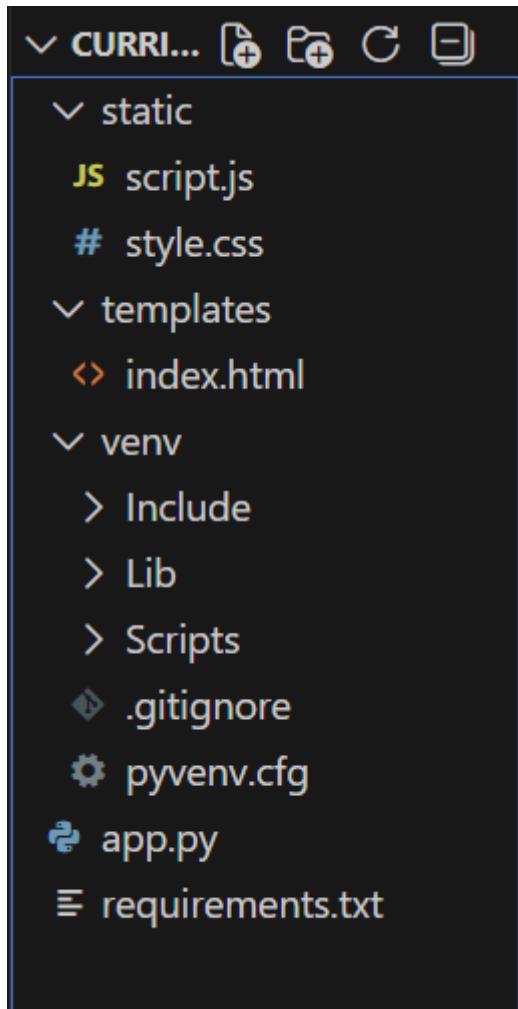
- Set up production environment
- Configure logging and error handling
- Document deployment steps
- Create user guide with screenshots

### Activity 5.2: Create Comprehensive Documentation

- Write API documentation
- Create user manual with examples
- Document configuration steps
- Create troubleshooting guide

## Technical Architecture

### Project Structure



## MILESTONE 1: Environment Setup & Ollama Configuration

This milestone establishes the complete local environment by installing Python, Flask, ReportLab, and Ollama with the IBM Granite 3.3 2B model. It verifies model connectivity, API response time, and system readiness for curriculum generation.

### Activity 1.1: Install Dependencies & Setup Environment

```
app.py > ...
1  from flask import Flask, render_template, request, jsonify
2  import requests
3  import json
4  import time
5  from reportlab.lib.pagesizes import letter
6  from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
7  from reportlab.lib.units import inch
8  from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Table, TableStyle
9  from reportlab.lib import colors
10 from io import BytesIO
11
```

### Step 2: Create requirements.txt

```
requirements.txt
1  Flask==2.3.0
2  requests==2.31.0
3  Werkzeug==2.3.0
4  python-dotenv==1.0.0
```

### Step 3: Create Virtual Environment

```
python -m venv venv
```

```
# Activate virtual environment
# Windows:
venv\Scripts\activate
# Mac/Linux:
source venv/bin/activate
```

**Step 4: Install Dependencies**

```
pip install -r requirements.txt
```

**Activity 1.2: Configure & Verify Ollama Setup****Install Ollama:**

1. Download Ollama from [ollama.ai](https://ollama.ai)
2. Follow installation instructions for your OS
3. Start Ollama service

**Pull Granite Model:**

```
ollama pull granite:3.3-2b
```

**Verify Installation:**

```
ollama list  
# Output should show: granite:3.3-2b
```

## MILESTONE 2: Core Backend Development

This phase builds the Flask backend, integrates the Ollama–Granite API, and implements the curriculum generation engine. It enables automated course creation, topic extraction, semester structuring, and PDF/JSON export functionality.

### Activity 2.1: Flask Application Setup

app.py:

```

❶ app.py > ...
1  from flask import Flask, render_template, request, jsonify
2  import reportlab.lib.styles could not be resolved from source Pylance(reportMissingModuleSource)
3  import (module) styles
4  import View Problem (Alt+F8) Quick Fix... (Ctrl+.) ⚡ Fix (Ctrl+)
5  from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
6  from reportlab.lib.units import inch
7  from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Table, TableStyle
8  from reportlab.lib import colors
9  from io import BytesIO
10
11 app = Flask(__name__)
12
13 # =====
14 # CONFIGURATION
15 # =====
16
17 MODEL_ID = "granite3.3:2b"
18 OLLAMA_API_URL = "http://localhost:11434/api/generate"
19
20 print("=" * 70)
21 print("🌐 Model:", MODEL_ID)
22 print("🌐 Endpoint:", OLLAMA_API_URL)
23 print("🤖 Running locally via Ollama")
24 print("=" * 70)
25
26 # =====
27 # CORS HEADERS
28 # =====
29

❷ app.py > ...
30
31 # =====
32 # OLLAMA INTEGRATION - OPTIMIZED FOR SPEED
33 # =====
34
35
36 def call_ollama_model(prompt):
37     """Call Ollama Granite 3.3:2b model to generate content"""
38     try:
39         response = requests.post(
40             OLLAMA_API_URL,
41             json={
42                 "model": MODEL_ID,
43                 "prompt": prompt,
44                 "stream": False,
45                 "temperature": 0.5 # Lower temperature for faster, more consistent responses
46             },
47             timeout=120 # Increased timeout
48         )
49
50         if response.status_code == 200:
51             result = response.json()
52             return result.get('response', '').strip()
53         else:
54             print(f"❗️ Ollama Error: {response.status_code}")
55             return None
56
57     except requests.exceptions.ConnectionError:
58         print("❗️ Could not connect to Ollama. Make sure Ollama is running: ollama serve")
59         return None
60     except requests.exceptions.Timeout:
61
62
63
64
65
66

```

```

⌚ app.py > ...
193  # =====
194  # CURRICULUM ENGINE - OPTIMIZED
195  # =====
196
197 def generate_curriculum_optimized(skill, level, semesters, weekly_hours, industry_focus):
198     """Generate curriculum with minimal API calls"""
199
200     print(f"\nGenerating {semesters} semesters of {skill} curriculum...")
201     start_time = time.time()
202
203     # Single optimized API call
204     curriculum_data = generate_complete_curriculum_batch(skill, level, semesters, industry_focus)
205
206     # Build final response
207     curriculum = {
208         "skill": skill,
209         "level": level,
210         "weekly_hours": weekly_hours or "20-25",
211         "industry_focus": industry_focus or "General Tech",
212         "semesters": curriculum_data.get("semesters", []),
213         "capstone_project": f"Complete {skill} Capstone Project"
214     }
215
216     elapsed = time.time() - start_time
217     print(f"Curriculum generated in {elapsed:.2f} seconds")
218
219     return curriculum
220
221 # =====

```

```

⌚ app.py > ...
222  # PDF EXPORT
223  # =====
224
225 def generate_pdf(curriculum):
226     """Generate professional PDF from curriculum"""
227     buffer = BytesIO()
228     doc = SimpleDocTemplate(buffer, pagesize=letter, rightMargin=0.5*inch,
229                             leftMargin=0.5*inch, topMargin=0.75*inch, bottomMargin=0.75*inch)
230
231     story = []
232     styles = getSampleStyleSheet()
233
234     # Title
235     title_style = ParagraphStyle(
236         'CustomTitle',
237         parent=styles['Heading1'],
238         fontSize=24,
239         textColor=colors.HexColor('#6366f1'),
240         spaceAfter=12,
241         alignment=1
242     )
243     story.append(Paragraph(f"{curriculum['skill'].title()} Learning Plan", title_style))
244
245     # Metadata
246     meta_style = ParagraphStyle('Meta', parent=styles['Normal'], fontSize=10, textColor=colors.grey)
247     meta_data = f"<b>Level:</b> {curriculum['level']} | <b>Weekly Hours:</b> {curriculum['weekly_hours']} | <b>Industry Focus:</b> {curriculum['industry_focus']}"
248     story.append(Paragraph(meta_data, meta_style))
249     story.append(Spacer(1, 0.3*inch))

```

```

❸ app.py > ...
377     # =====
378     # ROUTES
379     # =====
380
381     @app.route("/")
382     def index():
383         return render_template("index.html")
384
385     @app.route("/api/generate-curriculum", methods=["POST", "OPTIONS"])
386     def api_generate_curriculum():
387         """Generate curriculum using optimized Ollama call"""
388         if request.method == 'OPTIONS':
389             return jsonify({"status": "ok"}), 200
390
391         try:
392             data = request.json
393             skill = data.get("skill", "").strip()
394             level = data.get("level", "").strip()
395             semesters = data.get("semesters")
396             weekly_hours = data.get("weekly_hours", "").strip()
397             industry_focus = data.get("industry_focus", "").strip()
398
399             if not skill or not level:
400                 return jsonify({"error": "Skill and level are required"}), 400
401
402             try:
403                 semesters = int(semesters)
404                 if semesters < 2 or semesters > 8:
405                     raise ValueError
406
407             except ValueError:
408                 return jsonify({"error": "Semesters must be an integer between 2 and 8"}), 400
409
410             # =====
411             # MAIN
412             # =====
413
414             if __name__ == "__main__":
415                 print("\n" + "=" * 70)
416                 print("⚡ Optimized Curriculum Generator with Ollama Granite 3.3:2b")
417                 print("=" * 70)
418                 print("🌐 Flask server: http://localhost:5000")
419                 print("🔗 API: http://localhost:5000/api/generate-curriculum")
420                 print("❤️ Health: http://localhost:5000/health")
421                 print("=" * 70)
422                 print("⚠️ Make sure Ollama is running: ollama serve")
423                 print("=" * 70 + "\n")
424
425             app.run(host="0.0.0.0", port=5000, debug=True)
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
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2498
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2540
2541
2542
2543
254
```

## MILESTONE 3: Frontend Development

This milestone designs a modern, responsive interface for curriculum input and visualization using HTML, CSS, and JavaScript. It allows users to view generated courses, semesters, capstone details, and download PDF or JSON outputs.

## Create modern, responsive user interface

## Create templates/index.html:

```
templates > index.html > html > head
 1  <!DOCTYPE html>
 2  <html lang="en">
 3  <head>
 4      <meta charset="UTF-8">
 5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
 6      <title> GenAI Curriculum Generator</title>
 7      <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}>
 8  </head>
 9  <body>
10      <div class="container">
11          <!-- Header -->
12          <header class="header">
13              <div class="header-content">
14                  <h1> GenAI Curriculum Generator</h1>
15                  <p class="subtitle">Transform Skills into Semester-wise Syllabi</p>
16              </div>
17              <div class="header-accent"></div>
18          </header>
19
20          <!-- Main Content -->
21          <main class="main-content">
22              <!-- Input Section -->
23              <section class="input-section">
24                  <div class="section-header">
25                      <h2> Create Your Curriculum</h2>
26                      <p>Fill in the details to generate a comprehensive semester-wise syllabus</p>
27                  </div>
28
29                  <form id="curriculumForm" class="form-container">
```

```
templates > index.html > html > head
  2   <html lang="en">
  3     <body>
  4       <div class="container">
  5         <main class="main-content">
  6           <section class="input-section">
  7             <input type="text"
  8                   id="skill"
  9                   name="skill"
 10                   placeholder="e.g., Machine Learning, Full Stack Development"
 11                   required
 12             >
 13             <small>The main skill or technology you want to teach</small>
 14           </div>
 15
 16           <!-- Education Level -->
 17           <div class="form-group">
 18             <label for="level"> Education Level * </label>
 19             <select id="level" name="level" required>
 20               <option value="">Select Level</option>
 21               <option value="Diploma">Diploma</option>
 22               <option value="BTech">BTech / Bachelor's</option>
 23               <option value="Masters">Master's / Degree</option>
 24               <option value="Certification">Professional Certification</option>
 25             </select>
 26           </div>
 27
 28           <!-- Number of Semesters -->
 29           <div class="form-group">
 30             <label for="semesters"> Number of Semesters * </label>
```

```

templates > index.html > html > head
2   <html lang="en">
9     <body>
176       <script>
181         document.addEventListener('DOMContentLoaded', function() {
307           document.addEventListener('DOMContentLoaded', function() {
320             ...
326           if (!response.ok) {
328             throw new Error('Failed to generate PDF');
329           }
330
331           const result = await response.json();
332           const pdfBytes = new Uint8Array(result.pdf.match(/.../g).map(x => parseInt(x, 16)));
333           const blob = new Blob([pdfBytes], { type: 'application/pdf' });
334           const url = window.URL.createObjectURL(blob);
335           const a = document.createElement('a');
336           a.href = url;
337           a.download = `${window.currentCurriculum.skill}-curriculum.pdf`;
338           a.click();
339           window.URL.revokeObjectURL(url);
340
341           this.textContent = '⬇️ Download PDF';
342         } catch (error) {
343           console.error('Error downloading PDF:', error);
344           alert('Failed to download PDF: ' + error.message);
345           this.textContent = '⬇️ Download PDF';
346         } finally {
347           this.disabled = false;
348         }
349       </script>

```

```

templates > index.html > html > head
2   <html lang="en">
9     <body>
176       <script>
181         document.addEventListener('DOMContentLoaded', function() {
307           document.addEventListener('DOMContentLoaded', function() {
352             document.addEventListener('DOMContentLoaded', function() {
353               const jsonBtn = document.getElementById('downloadJsonBtn');
354               if (!jsonBtn) return;
355
356               jsonBtn.addEventListener('click', function() {
357                 if (!window.currentCurriculum) {
358                   alert('No curriculum to download');
359                   return;
360                 }
361
362                 const dataStr = JSON.stringify(window.currentCurriculum, null, 2);
363                 const dataBlob = new Blob([dataStr], { type: 'application/json' });
364                 const url = window.URL.createObjectURL(dataBlob);
365                 const a = document.createElement('a');
366                 a.href = url;
367                 a.download = `${window.currentCurriculum.skill}-curriculum.json`;
368                 a.click();
369                 window.URL.revokeObjectURL(url);
370               });
371             </script>
372           </body>
373         </html>

```

## MILESTONE 4: Deployment

This phase focuses on running the complete application locally by starting the Ollama service and Flask server. It ensures the system is accessible through a browser with stable performance.

### Activity 4.1: Local Deployment

#### Step 1: Start Ollama Service

```
ollama serve
```

#### Step 2: Run Flask Application

```
PS C:\Users\Surya\OneDrive\Desktop\curriculum_generator> python app.py
=====
💡 Model: granite3.3:2b
🌐 Endpoint: http://localhost:11434/api/generate
📍 Running locally via Ollama
=====

=====

🚀 Optimized Curriculum Generator with Ollama Granite 3.3:2b
=====

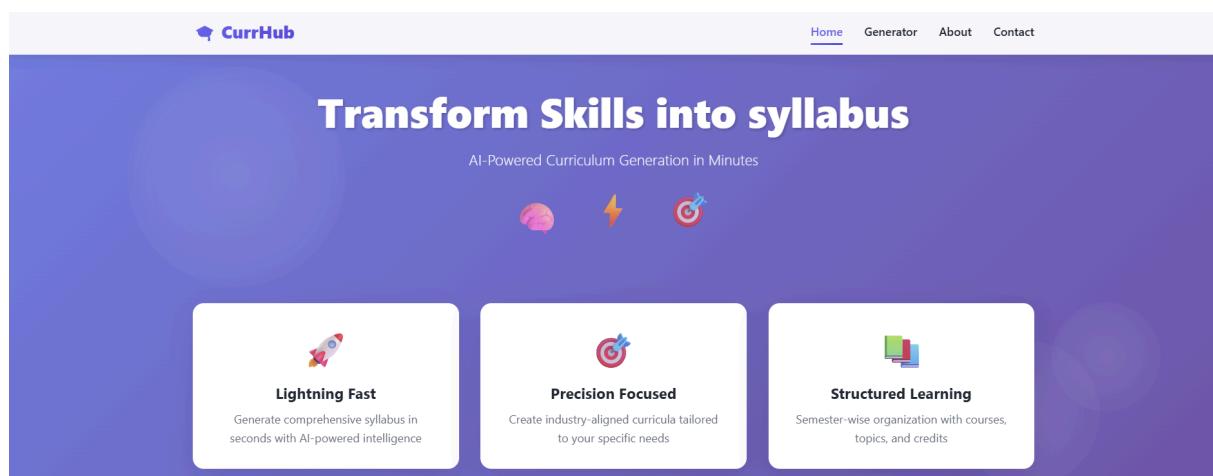
🌐 Flask server: http://localhost:5000
🔗 API: http://localhost:5000/api/generate-curriculum
❤️ Health: http://localhost:5000/health
=====

⚠️ Make sure Ollama is running: ollama serve
=====

* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
```

#### Step 3: Access Application

<http://localhost:5000>



The screenshot shows the CurrHub application interface. At the top, there is a navigation bar with the logo "CurrHub" and links for "Home", "Generator", "About", and "Contact". The main heading is "Transform Skills into syllabus" followed by the subtext "AI-Powered Curriculum Generation in Minutes". Below the heading are three circular icons: a pink brain-like icon, a yellow lightning bolt icon, and a red target icon. The bottom section features three white rectangular boxes with rounded corners, each containing an icon and text: "Lightning Fast" (rocket icon), "Precision Focused" (target icon), and "Structured Learning" (book icon). The background is a purple gradient with white circular highlights.

**CurrHub**

**Create Your Curriculum**  
 Fill in the details to generate a comprehensive semester-wise syllabus

**Skill**  
 e.g., Machine Learning, Web Development  
 The main skill or technology you want to teach

**Education Level**  
 Select Level

**Number of Semesters**  
 Select Duration

**Weekly Hours (Optional)**  
 e.g., 20-25  
 Hours per week dedicated to the course

**Industry Focus (Optional)**  
 e.g., AI, Web, Finance, Healthcare  
 Specific industry or domain focus

**Generate** **Clear**

**Build Your Vision**  
 Create comprehensive, industry-aligned curricula with AI  
 ✓ Export Ready

**CurrHub**

**About CurrHub**

CurrHub is an AI-powered curriculum generation platform that transforms educational skills into comprehensive, semester-wise syllabi in minutes.

Our mission is to empower educators, institutions, and organizations to create structured, industry-aligned learning paths without the hassle of manual curriculum design.

**Why Choose CurrHub?**

**Speed:** Generate complete curricula instantly with AI intelligence

**Quality:** Industry-aligned courses with real-world relevance

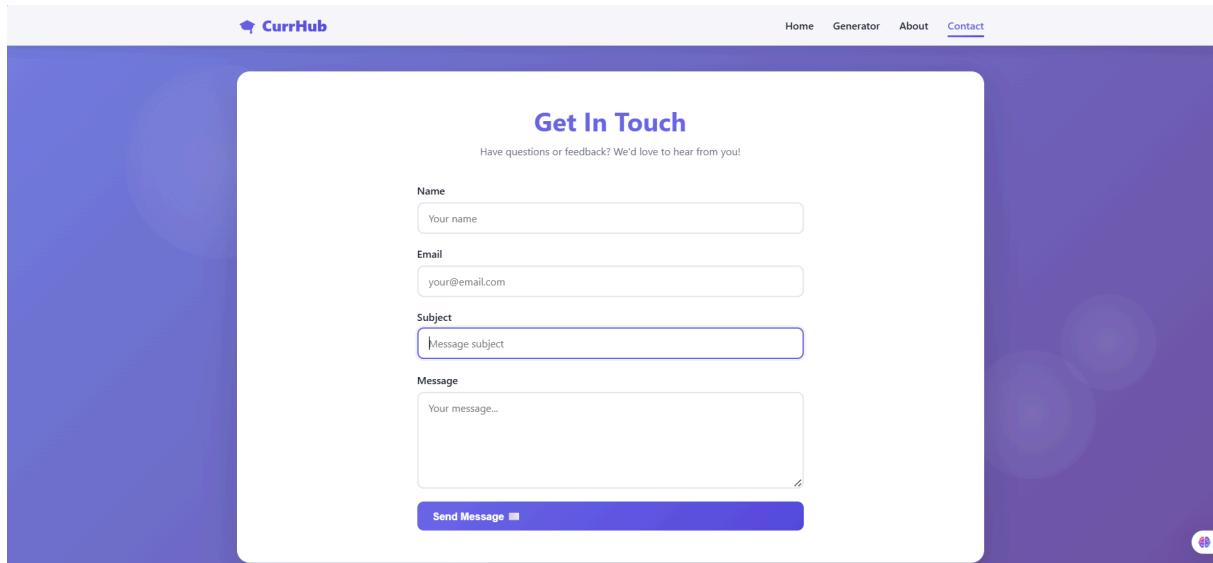
**Flexibility:** Customize for any duration, level, or domain

**Export Ready:** Download as PDF or JSON for immediate use

  
**AI-Powered**  
 Advanced LLM technology for intelligent curriculum design

  
**Lightning Fast**  
 Generate complete syllabi in seconds

  
**Precision**  
 Tailored to your specific educational needs



The screenshot shows a contact form titled "Get In Touch" on a purple-themed website. The form includes fields for Name, Email, Subject, and Message, each with placeholder text. A "Send Message" button is at the bottom, and a "Facebook" icon is in the bottom right corner.

**CurrHub**

Home Generator About Contact

## Get In Touch

Have questions or feedback? We'd love to hear from you!

Name

Email

Subject

Message

**Send Message**



## MILESTONE 5: Integration & Testing

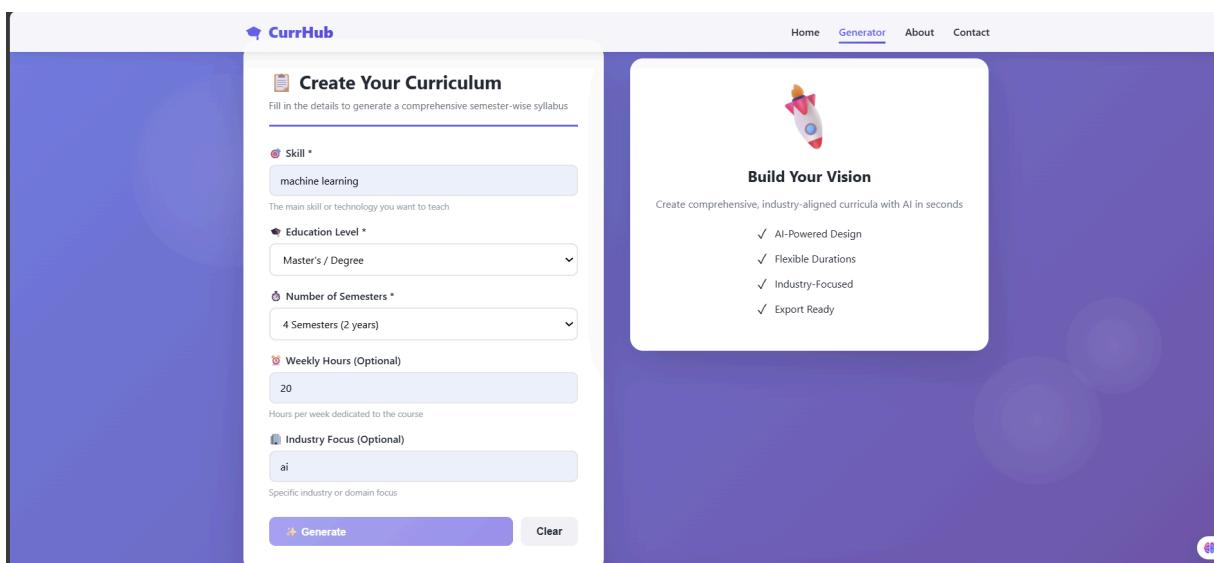
This milestone integrates all modules and performs functional, performance, and validation testing across different skills and education levels. It verifies AI response accuracy, PDF formatting quality, and end-to-end system reliability.

### Activity 5.1: Functional Testing

#### Test Case : Machine Learning Masters

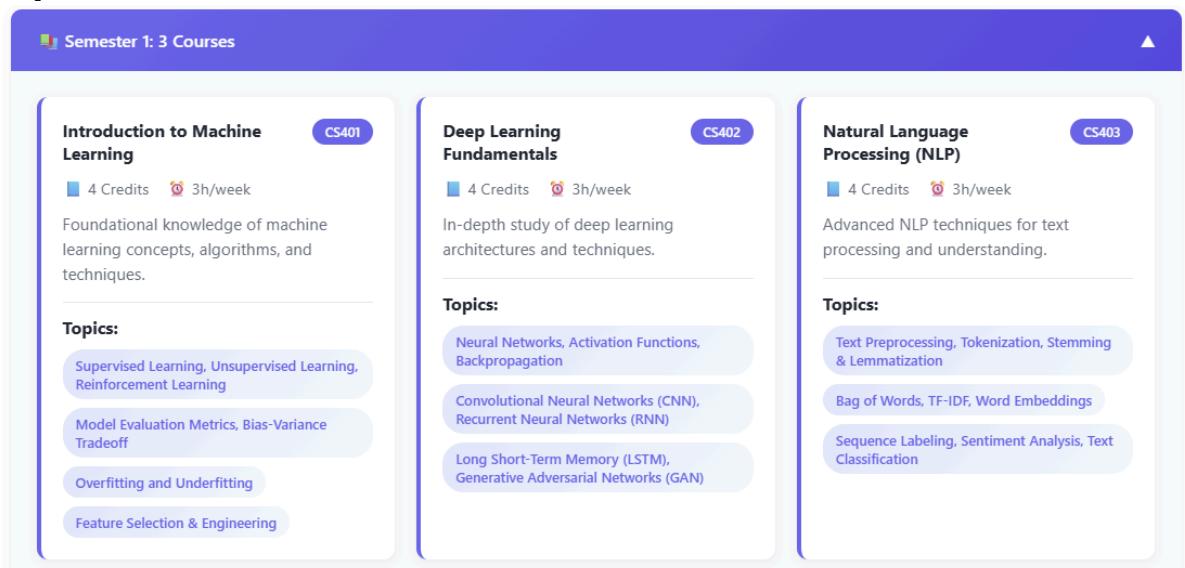
Input:

- Skill: Machine Learning
- Level: Masters
- Semesters: 4
- Weekly Hours: 20-25
- Industry Focus: AI



The screenshot shows the CurrHub platform. On the left, there is a form titled "Create Your Curriculum" with fields for Skill (Machine Learning), Education Level (Master's / Degree), Number of Semesters (4 Semesters (2 years)), Weekly Hours (Optional) (20), and Industry Focus (Optional) (ai). There are "Generate" and "Clear" buttons at the bottom. On the right, there is a promotional box titled "Build Your Vision" with the subtext "Create comprehensive, industry-aligned curricula with AI in seconds". It lists four features with checkmarks: AI-Powered Design, Flexible Durations, Industry-Focused, and Export Ready.

Out put :



The screenshot shows the generated curriculum for Semester 1: 3 Courses. It includes three courses: Introduction to Machine Learning (CS401), Deep Learning Fundamentals (CS402), and Natural Language Processing (NLP) (CS403).

- Introduction to Machine Learning (CS401)**  
 4 Credits, 3h/week  
 Foundational knowledge of machine learning concepts, algorithms, and techniques.  
**Topics:**
  - Supervised Learning, Unsupervised Learning, Reinforcement Learning
  - Model Evaluation Metrics, Bias-Variance Tradeoff
  - Overfitting and Underfitting
  - Feature Selection & Engineering
- Deep Learning Fundamentals (CS402)**  
 4 Credits, 3h/week  
 In-depth study of deep learning architectures and techniques.  
**Topics:**
  - Neural Networks, Activation Functions, Backpropagation
  - Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN)
  - Long Short-Term Memory (LSTM), Generative Adversarial Networks (GAN)
- Natural Language Processing (NLP) (CS403)**  
 4 Credits, 3h/week  
 Advanced NLP techniques for text processing and understanding.  
**Topics:**
  - Text Preprocessing, Tokenization, Stemming & Lemmatization
  - Bag of Words, TF-IDF, Word Embeddings
  - Sequence Labeling, Sentiment Analysis, Text Classification

### Semester 2: 3 Courses

**Machine Learning Systems Design** CS404

4 Credits 3h/week

Designing, deploying, and scaling machine learning systems.

**Topics:**

- Distributed Computing, MapReduce, Spark
- Cloud Platforms (AWS, Google Cloud, Azure)
- Data Pipelines and ETL Tools

**Advanced Deep Learning** CS405

4 Credits 3h/week

Exploring advanced deep learning architectures and optimization techniques.

**Topics:**

- Deep Convolutional Networks (DCN), RNN Architectures (LSTM, GRU)
- Transformers, BERT, GPT-2
- Neural Architecture Search (NAS)

**AI for Computer Vision** CS406

4 Credits 3h/week

Applying machine learning techniques to computer vision tasks.

**Topics:**

- Image Classification, Object Detection (YOLO, Faster R-CNN)
- Semantic Segmentation, 3D Object Detection
- Convolutional Neural Networks (CNN) for Computer Vision

### Semester 3: 3 Courses

**Reinforcement Learning** CS407

4 Credits 3h/week

Understanding and implementing reinforcement learning algorithms.

**Topics:**

- Markov Decision Processes (MDP), Q-Learning, SARSA
- Deep Reinforcement Learning (DRL)
- Policy Gradients, Actor-Critic Methods

**Ethics and AI** CS408

4 Credits 3h/week

Exploring the societal and ethical implications of AI.

**Topics:**

- Bias in Machine Learning, Fairness, Accountability
- Privacy Preservation Techniques (Differential Privacy)
- AI Governance, Ethical Frameworks

**AI for Time Series Analysis** CS409

4 Credits 3h/week

Applying machine learning techniques to time series data.

**Topics:**

- Time Series Forecasting (ARIMA, SARIMA)
- Prophet, LSTM for Time Series
- Anomaly Detection in Time Series

### Semester 4: 3 Courses

**Natural Language Processing Advanced Topics** CS410

4 Credits 3h/week

Advanced NLP techniques and deep learning applications.

**Topics:**

- Sentiment Analysis, Topic Modeling (LDA)
- Named Entity Recognition (NER), Part-of-Speech Tagging
- Deep Learning for NLP (Transformers)

**AI for Healthcare** CS411

4 Credits 3h/week

Applying machine learning to healthcare data and challenges.

**Topics:**

- Medical Image Analysis, Disease Diagnosis
- Predictive Modeling for Patient Outcomes
- Natural Language Processing in Electronic Health Records

**AI Project Workshop & Capstone** CS412

4 Credits 3h/week

Hands-on experience with designing, implementing, and presenting a machine learning project.

**Topics:**

- Project Proposal, Development, and Presentation
- Peer Review and Feedback
- Capstone Project Presentations

**PDF download :**

## Machine Learning Learning Plan

Level: Masters | Weekly Hours: 20 | Industry Focus: ai

### ■ Semester 1

Course Name	Code	Credits	Topics
machine learning - Course 1	MA101	4 Cr	Supervised Learning Algorithms, Unsupervised Learning Techniques, Model Evaluation Metrics, Feature Engineering, Cross-Validation
machine learning - Course 2	MA102	4 Cr	Deep Learning Fundamentals, Neural Network Architecture, Activation Functions, Backpropagation, TensorFlow Basics
machine learning - Course 3	MA103	4 Cr	Convolutional Neural Networks, Image Classification, Transfer Learning, Object Detection, CNN Applications

### ■ Semester 2

Course Name	Code	Credits	Topics
machine learning - Course 4	MA201	4 Cr	Recurrent Neural Networks, LSTM and GRU, Sequence Models, Natural Language Processing, Text Classification
machine learning - Course 5	MA202	4 Cr	Reinforcement Learning, Markov Decision Process, Q-Learning, Deep Q-Networks, Policy Gradient Methods
machine learning - Course 6	MA203	4 Cr	Advanced Topics, Model Optimization, Hyperparameter Tuning, Ensemble Methods, Real-world Applications

## Conclusion

**GenAI Curriculum Generator** successfully delivers an intelligent curriculum design assistant that combines advanced AI with educational expertise. By leveraging local Ollama deployment with Granite 3.3 2B, the system provides fast, accurate, and personalized curriculum recommendations without requiring cloud API keys. The platform demonstrates the viability of AI-powered curriculum design, making professional educational planning accessible to educators, institutions, and course designers while maintaining high performance, accuracy, and data privacy through local processing.