The Roadmap Generator - Project Report

Executive Summary

Project Name: The Roadmap Generator
Team ID: PNT2025TMID09555
Date: July 30, 2025
Category: Generative AI / Personalized Learning

The Roadmap Generator is an AI-powered web application that creates personalized, step-by-step learning roadmaps for any user-defined skill using the LLaMA2 language model. The application addresses the critical gap in personalized learning guidance by providing structured, free resource recommendations through an intuitive web interface.

1. Problem Definition & Market Analysis

1.1 Identified Problem Statements

The project addresses seven critical challenges in the learning ecosystem:

P1: Lack of Personalized Learning Paths

- Learners struggle to find structured, personalized roadmaps tailored to their current level, learning style, and desired skill
- Generic resources don't cater to individual knowledge levels or goals

P2: Overwhelming Volume of Learning Resources

- Vast sea of tutorials, courses, and books online without clear guidance on which to trust or follow
- Results in decision fatigue and abandoned learning attempts

P3: No Unified Skill Progress Tracker

- Absence of centralized platforms to track progress through skill-building steps with checkpoints and milestones

P4: Limited AI Integration in Learning Tools

- Traditional learning platforms don't leverage generative AI for dynamic content creation based on real-time user input

P5: Time-Consuming Manual Roadmap Creation

- Manual creation of skill-learning roadmaps is inefficient and prone to errors or inconsistency

P6: Lack of Free, Accessible Roadmaps

- Many curated roadmaps are locked behind paywalls, limiting access for budget-conscious learners

P7: Missing Integration Between Frontend UI and Smart Backend AI

- Learners need seamless interfaces where they can input a skill and receive clear, actionable AI-powered roadmaps

1.2 Target User Analysis

Primary Users:

- Students & freshers seeking guided skill learning
- Working professionals reskilling for tech roles
- Self-learners overwhelmed by unstructured online content
- Beginners uncertain about where to start with AI, Web Development, etc.

Key Pain Points:

- "I don't know which skill to learn next"
- "I'm confused by too many resources on the internet"
- "I waste time figuring out the right order of topics"
- "Free resources are scattered and hard to trust"
- "Generic roadmaps don't match my learning pace"

2. Solution Design & Architecture

2.1 Proposed Solution

The Roadmap Generator is a full-stack web application that:

- Accepts user input for desired skills through a Streamlit interface
- Processes input through a locally hosted LLaMA2 model via CTransformers
- Generates personalized learning roadmaps using advanced prompt engineering
- Recommends curated free resources in a structured format
- Displays results through an intuitive web interface

2.2 Problem-Solution Fit

Solution 1: Personalized AI-Powered Roadmap Generation

- Uses LLaMA2's contextual understanding to generate tailored learning sequences
- Saves time, removes confusion, increases motivation through clear goals

Solution 2: Curated Free Resource Integration

- Includes references to high-quality, free resources (MOOCs, blogs, YouTube, GitHub repos)
- Ensures affordability and accessibility without compromising quality

Solution 3: End-to-End Web Application

- Streamlit-based interface with LLaMA2 backend provides one-stop solution
- Lightweight, fast, scalable, and deployable without cloud dependencies

2.3 System Architecture

[User Input] → [Streamlit UI] → [Backend: Python + CTransformers]

→ [LLaMA2 Inference] → [Roadmap Output] → [Streamlit Display]

Architecture Components:

1. Frontend - Streamlit
    - User-friendly interface for input/output
    - Handles skill input and roadmap display
2. Backend - Python Application
    - Streamlit-powered backend
    - Manages logic and LLaMA2 integration
3. Model Integration - CTransformers + LLaMA2
    - Uses quantized .gguf version of LLaMA2
    - Responds to custom prompts for learning path generation
4. Model Setup - setup.sh
    - Auto-downloads model from Hugging Face
    - Ensures reproducibility and easy deployment

2.4 Technology Stack

Layer	Technology	Purpose
Frontend	Streamlit	UI for user inputs and roadmap display
Backend	Python + Streamlit	API logic and model integration

Model Integration	CTransformers	Interface for .gguf LLaMA2 models
Model File	LLaMA2 (GGUF Format)	Pre-trained text generation model
Environment	requirements.txt + venv	Package management
Model Setup	setup.sh	Automated model download
Deployment	Streamlit Cloud	App hosting and access
Version Control	Git + GitHub	Code repository management

3. Project Planning & Development

3.1 Product Backlog

ID	Feature/Task	Priority
PB1	User Input UI for skill entry	High
PB2	Backend setup with Streamlit	High
PB3	Integrate LLaMA2 using CTransformers	High

PB4	Prompt creation engine for roadmap generation	High
PB5	Fetch & display AI-generated roadmap	High
PB6	Resource recommendation integration	Medium
PB7	Styling and frontend polishing	Medium
PB8	Save/Download roadmap feature	Low
PB9	Deployment on Streamlit Cloud	High
PB10	Model auto-download with setup.sh	High
PB11	Documentation & README for GitHub	High
PB12	Progress tracker or milestone badges	Low

3.2 Sprint Planning

Sprint	Duration	Goal

Sprint 1	5 days	Core functionality: UI + AI Integration
Sprint 2	4 days	Polish UI, add features like download/save
Sprint 3	3 days	Deployment, testing, documentation

3.3 User Stories & Story Points

Story ID	As a...	I want to...	So that I can...	Story Points
US1	user	enter a skill into a web form	get a personalized roadmap	3
US2	system developer	use Streamlit to connect UI with backend	build the app quickly	2
US3	system	send prompts to LLaMA2 via CTransformers	receive meaningful roadmap content	5
US4	learner	view recommended free resources	get started without manual searching	3

US5	user	download my roadmap as PDF	access it offline	2
US6	developer	deploy app to Streamlit Cloud	make it accessible online	3
US7	contributor	run setup.sh to get the model	avoid uploading large files manually	2
US8	user	track completed steps	stay organized while learning	5

4. Data Flow & System Design

4.1 Data Flow Diagram

Level 0: Context Diagram

[User] --Input Skill--> [Roadmap Generator] --Roadmap + Resources--> [User]

Level 1: Detailed Flow

User Input → Streamlit Frontend → app.py Backend → Model Loader

→ LLaMA2 Model → Streamlit Renderer → User Display

4.2 Folder Structure

```
roadmapper/
├── app.py              # Main Streamlit application
├── setup.sh            # Model download script
├── requirements.txt    # Python dependencies
├── README.md           # Project documentation
├── .gitignore          # Ignore large model files
├── utils/
│   └── model_loader.py   # LLaMA2 integration code
└── model/
    └── llama-2-7b-chat.Q3_K_M.gguf  # Auto-downloaded model
```

5. Requirements Specification

5.1 Functional Requirements

1. Skill Input Interface
   - Web UI for skill entry with optional auto-suggestions
   - Support for various skill domains (AI, Web Dev, Data Science, etc.)
2. Roadmap Generation
   - Process input using LLaMA2 model via CTransformers
   - Generate personalized, step-by-step roadmaps with:
     - Learning objectives
     - Duration estimates
     - Free resource recommendations
3. Model Integration
   - Load LLaMA2 .gguf model efficiently
   - Process prompts and parse outputs effectively
4. Web Interface
   - Clean roadmap display via Streamlit
   - "Generate Again" functionality
   - Optional PDF export capability
5. Model Management
   - Automated model download via setup.sh
   - Efficient model loading and caching

5.2 Non-Functional Requirements

1. Performance
   - Roadmap generation within 10-15 seconds
   - No crashes for typical user inputs
2. Scalability
   - Support multiple simultaneous users
   - Modular design for future enhancements
3. Reliability
   - Robust error handling
   - Consistent model availability
4. Maintainability

    - Modular code structure
    - Comprehensive documentation
5. Security
    - Input validation to prevent prompt injection
    - No logging of sensitive data
6. Portability
    - Compatible with Python 3.8+
    - Minimal setup requirements

6. Testing & Quality Assurance

6.1 User Acceptance Testing Framework

Testing Environment: Streamlit Cloud/Local
Model: LLaMA2 7B Chat (via CTransformers, GGUF format)
Testing Tools: Streamlit Community, Streamlit load tester, Locust

6.2 Test Features

Feature ID	Feature	Acceptance Criteria
F1	Skill Input	User can input any skill name
F2	Roadmap Generation	Model returns step-by-step roadmap with resources
F3	UI Responsiveness	Streamlit UI is intuitive and responsive
F4	Loading Time	Roadmap generation completes in <10 seconds
F5	Error Handling	Handles invalid input gracefully

F6	Model Integration	LLaMA2 inference works as expected
F7	Resource Suggestions	Includes high-quality, free learning resources

6.3 Test Cases

Functional Tests:

- TC01: Skill input accepts valid data (Pass)
- TC02: Handles empty input appropriately (Pass)
- TC03: Generates multi-step roadmaps (Pass)
- TC04: Provides accessible resource links (Pass)
- TC05: Meets response time requirements (Pass)

Performance Tests:

- PT01: Handles concurrent usage without crashes (Pass)
- PT02: Model cold start under 30 seconds (Pass)
- PT03: Processes complex skill names effectively (Pass)
- PT04: UI loads within 2 seconds (Pass)

7. Implementation Details

7.1 Key Implementation Features

1. Prompt Engineering
 prompt = f"""You are an expert roadmap generator.
2. Create a detailed step-by-step learning path to master: {user_input}.
3. Include tools, free resources, and time estimates."""
4.
5. Model Loading
    - Efficient .gguf model loading via CTransformers
    - Memory optimization for CPU inference
    - Caching to avoid reloading
6. Resource Integration
    - Curated free resources (YouTube, Coursera, GitHub, MOOCs)
    - Quality filtering and relevance ranking
    - Structured presentation by learning phase

7.2 Deployment Strategy

1. Development Setup
    - Clone GitHub repository
    - Run setup.sh for model download
    - Launch with streamlit run app.py
2. Production Deployment
    - Streamlit Cloud integration
    - Automated CI/CD via GitHub
    - Lightweight repository (no large model files)

8. Future Enhancements & Roadmap

8.1 Phase II Enhancements

1. User Authentication & Personalization
    - User accounts for saved roadmaps
    - Learning progress tracking
    - Personalized recommendations
2. Advanced Input Options
    - Difficulty level selection
    - Timeline preferences
    - Learning style customization
3. Enhanced Resource Management
    - Local embeddings for resource ranking
    - User ratings and feedback system
    - Community-contributed resources
4. Export & Integration Features
    - Calendar synchronization
    - PDF/Markdown export
    - Learning management system integration

8.2 Technical Improvements

1. Performance Optimization
    - GPU acceleration support
    - Model quantization improvements
    - Caching layer implementation
2. Scalability Enhancements
    - Multi-model support
    - Load balancing capabilities
    - Database integration for user data

9. Project Outcomes & Success Metrics

9.1 Delivered Components

1. ✅ Functional Streamlit web application

2. ✅ LLaMA2 integration via CTransformers
3. ✅ Automated model setup system
4. ✅ Comprehensive documentation
5. ✅ Testing framework and validation
6. ✅ Deployment-ready architecture

9.2 Key Achievements

- Problem-Solution Fit: Successfully addresses all identified learning challenges
- Technical Innovation: Effective integration of open-source LLM with web framework
- User Experience: Intuitive interface with fast response times
- Scalability: Modular architecture supporting future enhancements
- Accessibility: Free, open-source solution with no paywalls

9.3 Impact Assessment

For Learners:

- Reduced time to find quality learning resources
- Clear, structured learning paths
- Access to free, curated educational content
- Personalized learning experience

For Educators:

- Tool for creating structured curricula
- Resource for guiding student learning paths
- Framework for skill development programs

For Organizations:

- Employee upskilling and reskilling support
- Standardized learning path creation
- Cost-effective training resource

10. Conclusion

The Roadmap Generator successfully demonstrates the power of combining generative AI with intuitive web interfaces to solve real-world learning challenges. By leveraging LLaMA2's language understanding capabilities through a user-friendly Streamlit application, the project creates a valuable tool for learners across various domains.

The solution addresses critical gaps in personalized learning guidance while maintaining accessibility through free resources and open-source technology. The modular architecture and comprehensive testing framework ensure reliability and scalability for future enhancements.

This project represents a significant step toward democratizing personalized education and showcases the practical applications of large language models in educational technology.

Project Team: PNT2025TMID09555
Report Prepared By: Bhagyashri Sandeep Badhekar
Date: July 30, 2025
Total Project Marks: 25 Marks (Across all phases)