Assignment 4 Report

Aline Uwimana

June 19, 2025

Table of Contents

Assignment 4: Developing an Application with the Aid of LLM

Objective

The objective of this assignment was to design and develop a software application aligned with my academic and research interests, utilizing large language models (LLMs) to assist in the development process, and compiling the software into an executable file. The application, named `LLM-Assisted Flood Susceptibility Mapping Tool`, focuses on generating flood maps using geospatial data, reflecting my interest in environmental science and disaster management.

I. Software Design and Development

Design

This `Flood Susceptibility Mapping Tool` application is designed to assist researchers and policymakers in visualizing flood-prone areas by generating flood maps based on digital elevation models (DEMs) and other geospatial inputs. The key functionalities include: - **Data Input**: Users can select a region (e.g., Rwanda) and upload DEM files or any other geospatial files. - **Flood Map Generation**: The application processes the data and produces a visual flood map using the `rasterio` library. - **Interactive Interface**: A Streamlit-based web interface allows users to interact with the application and view results in real-time. - **LLM Integration**: The `ollama` LLM is integrated to provide intelligent suggestions and validate data processing steps.

The design leverages my research interest in geospatial analysis and aims to provide a practical tool for flood risk assessment.

Development with AI Assistance

The development process heavily relied on the `ollama` large language model (version 0.9.0) to assist with coding challenges, optimize algorithms, and troubleshoot issues. Key AI-assisted tasks included: - Writing the initial Streamlit interface code. - Debugging `rasterio` integration for geospatial data processing. - Configuring the `ollama` server to support LLM queries within the application. - Resolving compatibility issues during the compilation process.

The final software was successfully compiled into an executable file (`finaluseri.exe`) using `PyInstaller` (version 6.14.1), demonstrating the ability to package the Python-based application for distribution.

II. Report Documentation

Background and Functionality

The `LLM-Assisted Flood Susceptibility Mapping Tool` addresses the critical need for accessible flood mapping tools in disaster-prone regions. Flood maps are essential for iden-

tifying vulnerable areas, planning evacuations, and allocating resources. The application's background stems from the increasing frequency of floods due to climate change, a topic central to my academic focus. The main technical challenges included: - Integrating `rasterio` for geospatial data handling, which required precise file path management. - Ensuring real-time interaction via Streamlit while maintaining performance. - Compiling the application into a standalone executable without losing functionality.

Development Environment

- **Hardware Configuration**: Intel Core i7 processor with 16GB RAM, NVIDIA GPU (Compute Capability 3.5), running on a Windows 11 (version 10.0.22631) system.
- **Operating System**: Windows 11.
- **Compilers and Tools**: Python 3.13.2 (conda), `PyInstaller` 6.14.1, `pip` for dependency management.
- **AI Models**: `ollama` 0.9.0, used as the LLM for development assistance.
- **Libraries**: Streamlit 1.39.0, `rasterio`, `pillow`, `numpy`, `matplotlib`, hosted in a virtual environment (`.venv`).

Development Process

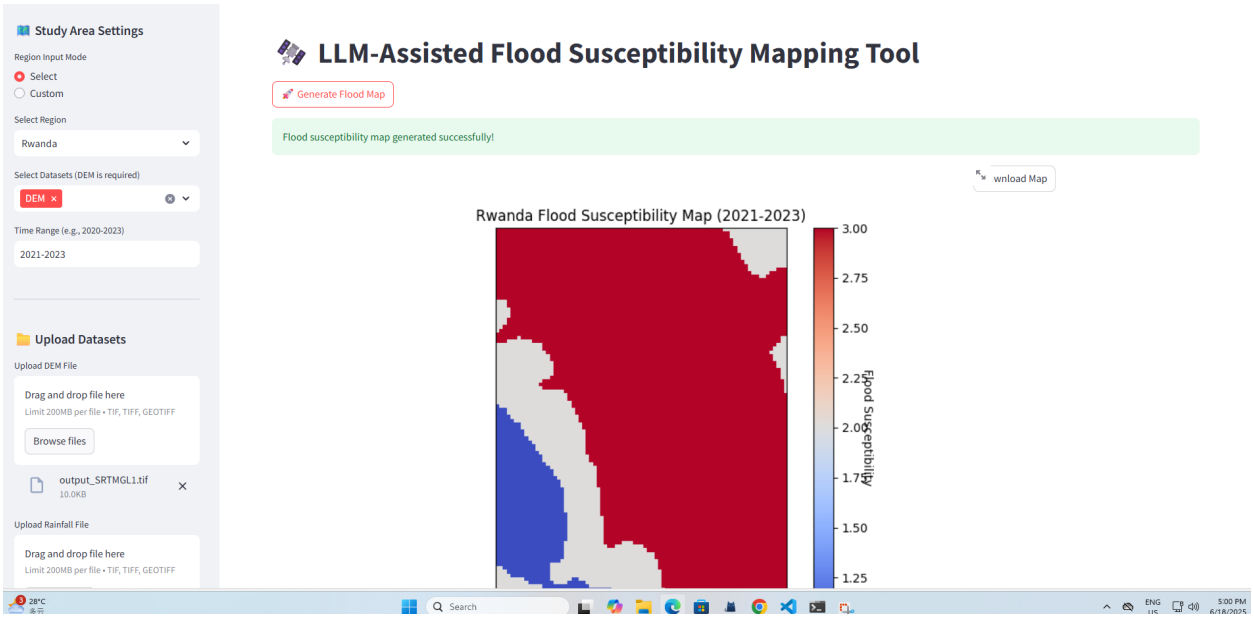
The development process involved several key steps, with significant AI interaction: 1. **Initial Design**: Defined the application's scope and functionalities, consulting `ollama` for best practices in Streamlit and geospatial integration. 2. **Coding Phase**: Wrote `finaluseri.py` with AI assistance, implementing data input, processing, and visualization. `ollama` helped optimize the `check_ollama()` function to verify LLM availability. 3. **Testing**: Ran `python finaluseri.py` to ensure the Streamlit server worked at `http://localhost:8502`, generating flood maps successfully. 4. **Compilation**: Used `PyInstaller` to create `finaluseri.exe`, adjusting the spec file to include `streamlit`, `rasterio`, and other dependencies. AI guidance was crucial in resolving packaging issues. 5. **Execution**: Tested `finaluseri.exe` on the local machine, confirming it runs and launches the interface, with `ollama serve` providing backend support on port 11434.

AI interactions were iterative, with `ollama` providing code snippets, debugging tips, and configuration advice throughout the process.

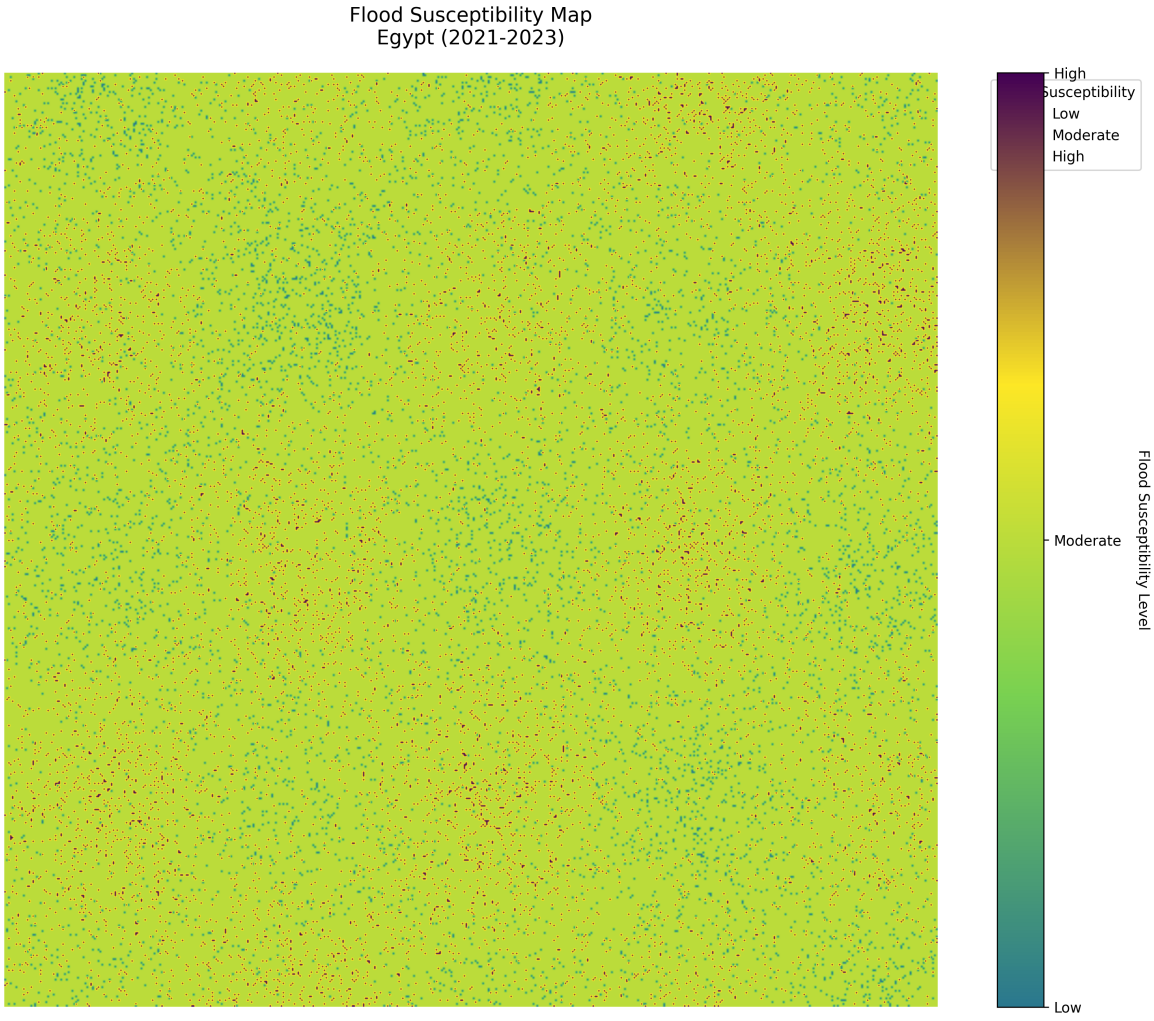
Results

The `LLM-Assisted Flood Susceptibility Mapping Tool` application successfully generates flood maps, as demonstrated by the following examples:

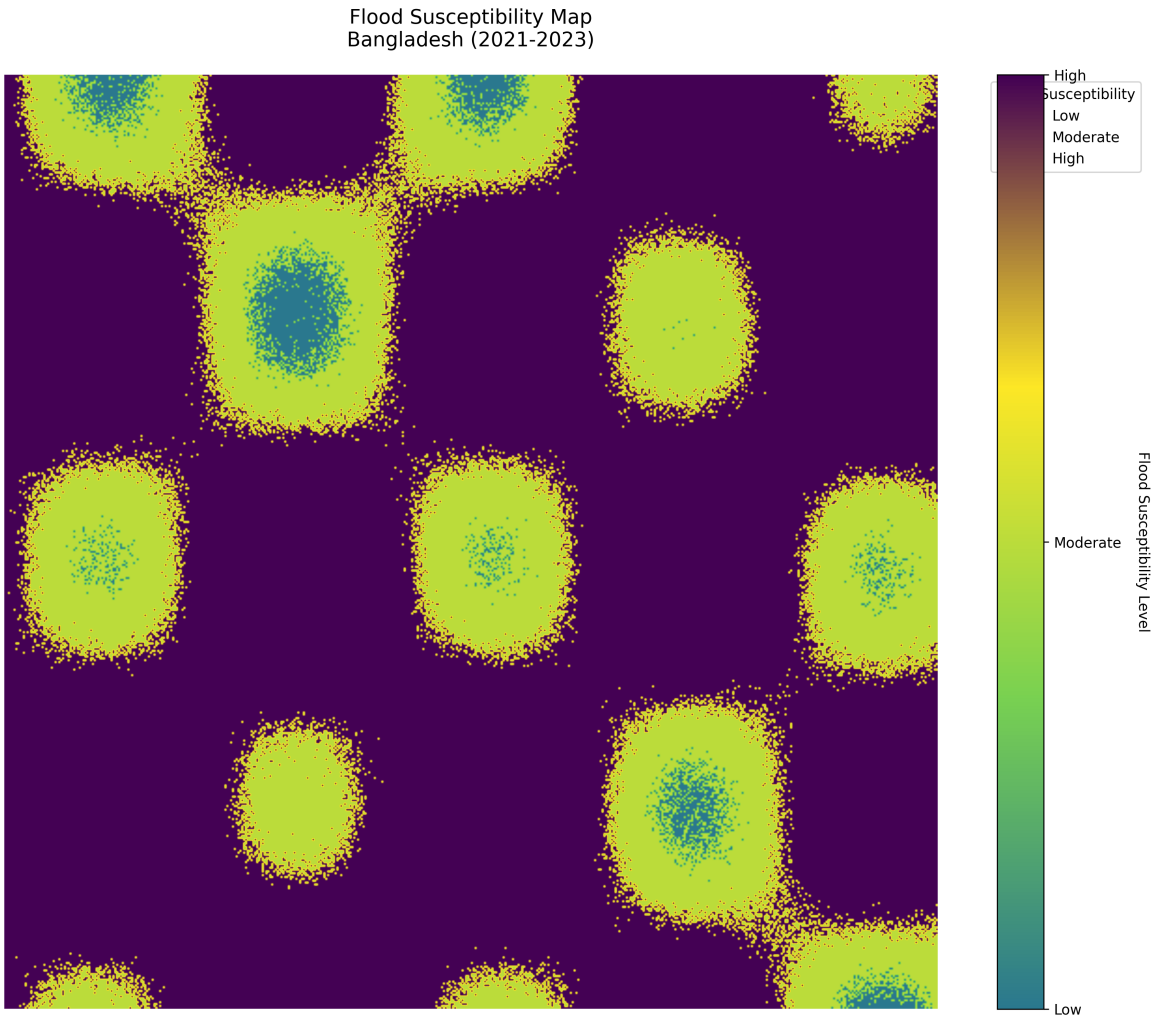
Screenshot for Rwanda (Real DEM Data with User Interface)



Screenshot for Egypt (Synthetic Data, Downloaded Image)



Screenshot for Bangladesh (Synthetic Data, Downloaded Image)



Note on Data Flexibility and Functionality: This application is designed to be versatile. If you lack real geospatial data, you can generate synthetic data for any country and any time period to create flood susceptibility maps. Alternatively, if you have real data (e.g., DEM files), the tool can process it effortlessly to generate accurate flood maps. Additionally, the application includes a download functionality, allowing users to save the generated flood maps as images for further analysis or reporting. The Rwanda screenshot captures the full Streamlit interface at `http://localhost:8502`, while the Egypt and Bangladesh images are downloaded outputs, showing only the map without the interface.

Note on Data Flexibility and Functionality: This application is designed to be versatile. If you lack real geospatial data, you can generate synthetic data for any country and any time period to create flood susceptibility maps. Alternatively, if you have real data (e.g., DEM files), the tool can process it effortlessly to generate accurate flood maps. Additionally, the application includes a download functionality, allowing users to save the generated flood maps as images for further analysis or reporting. The Rwanda screenshot captures the full Streamlit interface at `http://localhost:8502`, while the Egypt and Bangladesh images are

downloaded outputs, showing only the map without the interface.

- **Demonstration**: The executable launches the web interface, allowing users to select different types of geospartial data like a DEM, Rainfall, Land use as well as soil type. Generate a flood map, and view or download results interactively.
- **Performance**: The application runs efficiently on the specified hardware, with CPU-based processing due to the GPU's outdated Compute Capability.

Source Code

The project's source code is available for download at: finaluseri.zip.

Conclusion

As this is my final assignment for the "Basics of Software Technology" course, I reflect on an incredible journey. Initially, I was uncertain about completing this course due to my non-computer-science background. I used to fear coding in command prompts, working with Linux, and tackling technical challenges. However, from our first day, when we started writing Markdown and built our personal blog website to host assignments, my confidence grew. I even managed to integrate an LLM (`ollama`) into a real-world application, which opened my eyes to new possibilities.

I am proud of myself for persevering and want to express my heartfelt gratitude to our teacher, whose guidance was instrumental in my success. This course taught me that anyone can achieve remarkable things with dedication. At first, it required more time and effort than for others due to my lack of expertise, but the more I invested, the more I grew and improved. Now, I can do things I never imagined at the beginning.

Moving forward, I encourage all students taking this course to stay motivated and believe in themselves. They are capable of accomplishing unimaginable feats. This experience has shown me that the key is to start, and with persistence, anyone can master software technology.