**AI Is Eating Your Algorithms**

How simple prompt engineering can replace custom software

There is a lot of excitement surrounding the [new software](https://archive.is/o/ONLFj/https:/openai.com/index/hello-gpt-4o/) made possible by generative ai. Applications like natural language chatbots that respond intelligently to complex questions with custom knowledge pique interest because these applications were previously impossible. Generative ai has very quickly enabled a whole new set of applications, and this is justifiably exciting.

Less exciting but equally impactful will be the replacement of traditional, custom-developed software with general purpose LLMs to solve software problems that we could already solve, but can now be implemented without problem-specific code. Generative ai is highly flexible and with a bit of prompt engineering a large language model (such as GPT-4o) can be used to solve a significant number of traditional software challenges. While it is exciting to build new things previously impossible (or highly infeasible), much of the impact of generative ai will be building differently what we can already build today.

This shift from problem-specific application development to integrating problem-agnostic, generalist generative ai will change dominant architecture patterns, organizational skill set requirements, and cost-of-development equations. It is the reason why every development organization should understand the abilities (and limitations!) of generative ai, even if not building shiny new generative ai applications.

Many underestimate the volume of traditional software that can be replaced by generalized solutions. Daily I find myself thinking… “this could really just be a prompt”.

Below are just a few examples. They are not necessarily the most technically impressive or societally impactful, but should show how much software could be replaced by some type of generative ai.

**Pond Water Level**

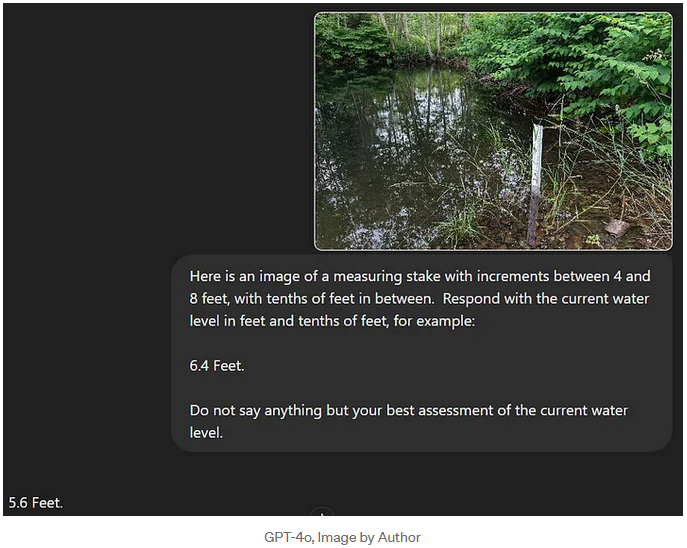
I have a pond. The water level in the pond varies significantly based on season and weather. I have always wanted to digitally measure and track the water level over time.

[Measuring water level](https://archive.is/o/ONLFj/https:/www.ysi.com/parameters/level%23how-to-measure-water-level) continuously and remotely requires a gauge that must translate a measurement into a digital recording to be interpreted by custom software. The tools that do this (as run by the USGS to measure river flow) require installation and aren’t cheap. They are custom software-hardware solutions for the specific problem of water level measurement and required effort and engineering to build.

I don’t have those engineering skills nor do I want to spend the money to buy an expensive digital gauge. Here’s a different solution, made possible by generative ai:

1. Point a web-connected camera at the pond (I already have this)
2. Put a graduated stick in the pond
3. Send the image to a multi-modal LLM (GPT-4o) and simply ask “how high is the water”
4. Parse and record the answer.

No custom gauges. Little custom code, just some plumbing to tie the pieces together. Here is an image of the pond and GPT-4o’s assessment of the level via the chat interface.

GPT-4o, Image by Author

Obviously this is just the chat interface, and code still needed to be written to obtain the image from the camera, send it to the GPT-4o URL, parse the response, and record it. But compared to building a digital water monitoring gauge, this was a simple engineering problem.

Is the LLM solution as efficient and reliable as the custom water gauge? Probably not. I expect snow will confuse the LLM. While the camera has IR, very low light situations cause issues. GPT-4o, despite explicit instructions, doesn’t always respond in the way requested, breaking parsing and forcing a retry. The measurement interval is limited by my OpenAI subscription, and OpenAI does experience downtime. There are probably other drawbacks to the LLM solution that I haven’t considered.

But, it could be implemented quickly with little cost. And with the continuous improvement of models, it won’t be long before the image evaluation could be performed locally and for free, eliminating the need for a remote call to GPT-4o.

Water level measurement is not exactly a sexy use case, but shows how easily an LLM can replicate functionality that previously required a custom solution.

**Global Forecast Descriptions**

Many weather sites (Weather Underground, AccuWeather, etc.) provide human readable descriptions of the raw forecast data generated by the Global Forecasting System run by the National Weather Service. However, because of the sheer number of locations and times, these long-form forecasts must be algorithmically generated.

For example, here is the forecast for one specific time and place (June 20th near Denver, CO):

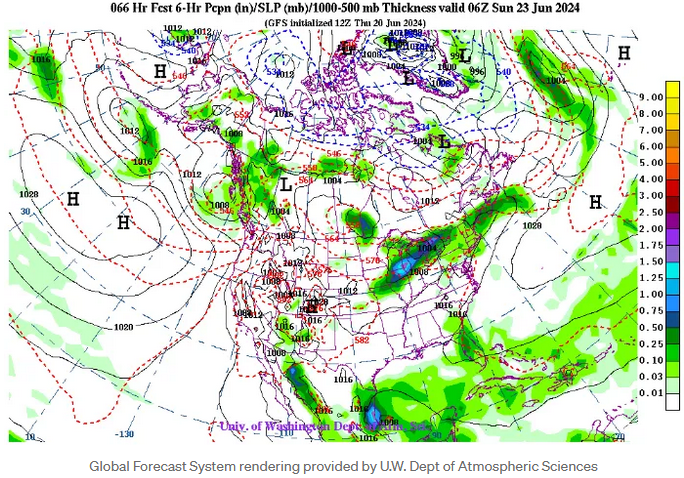
*A 50 percent chance of showers and thunderstorms, mainly before midnight. Mostly cloudy, with a low around 60. Breezy, with a south wind 15 to 20 mph decreasing to 9 to 14 mph in the evening. Winds could gust as high as 34 mph.*

And one for southern Texas on in July:

*Hurricane conditions possible. Showers likely and possibly a thunderstorm before 7pm, then showers and thunderstorms likely between 7pm and 10pm, then showers likely and possibly a thunderstorm after 10pm. Mostly cloudy, with a low around 77. Chance of precipitation is 60%.*

These generated descriptions can be quite varied depending on the forecast, Sometimes they highlight different aspects (wind, temperature, precipitation, etc.), and scan significantly changes the tone, order, and format of the description. If we wanted to implement this today, would we code this complex algorithm, or could we do it faster with the existing abilities of an LLM?

To give you an idea, here is GPT-4o’s prediction for Seattle, WA from a single image provided by GFS:



Global Forecast System rendering provided by U.W. Dept of Atmospheric Sciences

*Based on the weather map provided, Seattle, WA is expected to experience the following weather conditions:*

* *There is a low-pressure system (L) located offshore to the west of Seattle.*
* *This low-pressure system is associated with a significant amount of precipitation, as indicated by the green shading, which suggests rain.*
* *The central pressure of the low is around 997 mb, indicating a fairly strong system.*
* *The pressure gradient is relatively tight, suggesting the possibility of windy conditions.*
* *Given the proximity of the low-pressure system, Seattle can expect rain and potentially windy conditions on Friday, June 28, 2024.*

*In summary, Seattle, WA is likely to have rainy and windy weather on June 28, 2024, due to the approaching low-pressure system from the west.*

Now, this forecast was wrong. June 28th was a beautiful sunny day, and the suggested rain did not hit until later. However, consider the understanding that the LLM demonstrated with this response: it knows where Seattle, WA is on that map, it knows what the green color represents, it knows what pressure isobars are and how they affect weather, it knows that weather systems move from west to east, it was able to read the date of the top without being explicitly told, etc. All this despite the fact it was never specifically trained to interpret weather data.

If we had access to the underlying data used in that image and a reasonably powerful LLM, we could generate accurate long-form forecast descriptions with only basic prompt engineering, replicating or exceeding whatever custom algorithm is currently creating them. Future use cases that describe data, regardless of how complex, will be implemented by a prompt over a general purpose LLM, not code custom written for the specific use case.

**Plant Pathology Identification**

I have an unknown type of cherry tree that periodically suffers from some type of leaf disease. I’m a new orchardist, so I have no clue what’s afflicting the tree, nor what I should do about it.

There are applications that can identify plants and diseases. These applications usually cost money to use, because their implementation cost money to build.

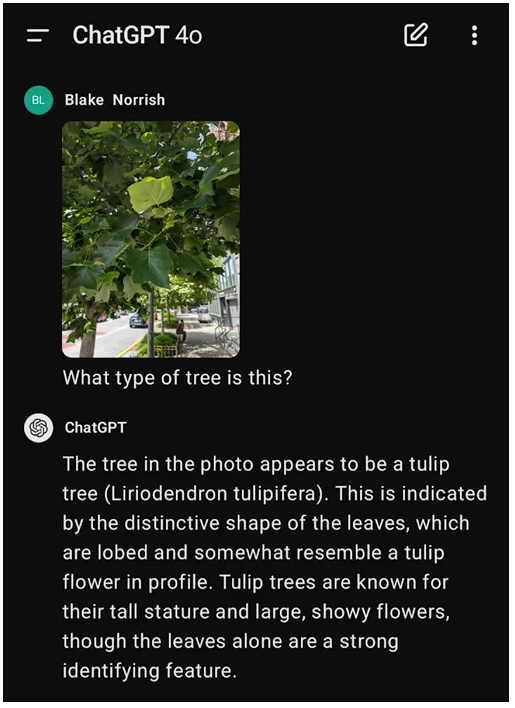
For example, PictureThis is the most popular plant identification app on the Android Play Store with 100M+ users. It costs $2.99 for the first month, then $49.99 per year.

Want plant identification and diagnosis? Pay for it. Image by Author

PictureThis is *already* anAI/ML based application. However, unlike GPT-4o, it’s a custom solution trained on a specific data set to answer a specific type of question.

GPT-4o was not trained to identify plants or plant diseases, surely it can’t replicate and replace all the hard work and engineering put into PictureThis can it?

As a warm up test, here’s GPT-4o identifying a tree I’ve admired on the way to work (it was correct).



GPT-4o giving me exaclty what I’m looking for. Image by Author.

Let see how it does with my diseased cherry tree:

*This is an image of a cherry tree leaf taken in Western Washington in June. Given this information, what affects the tree?*



Diseased Cherry Tree. Image by Author.

*The cherry tree leaves in the image show signs of a common problem that affects cherry trees: leaf spot disease, specifically Cherry Leaf Spot caused by the fungus Blumeriella jaapii. The small, circular spots that turn reddish to purple on the leaves are typical symptoms of this disease. Over time, the spots may enlarge, and the affected leaves can turn yellow and drop prematurely.*

*Here are a few steps you can take to manage and prevent this disease:*

1. ***Sanitation:*** *Remove and dispose of fallen leaves and infected plant debris to reduce sources of infection.*
2. ***Fungicide Application:*** *Use appropriate fungicides during the growing season, especially in the spring when new leaves are emerging.*
3. ***Pruning:*** *Improve air circulation by pruning the tree to open up the canopy, which can reduce the humidity levels that favor fungal growth.*
4. ***Water Management:*** *Avoid overhead watering to minimize the leaf wetness period, which is critical for fungal infections.*

*If the problem persists, consulting a local agricultural extension service or a professional arborist can provide more targeted advice for managing cherry leaf spot in your specific region.*

This is an accurate and detailed response for software that was not built for this use case.

Think of it this way: if you were a start-up investor, and someone approached you looking for money to build an app like PictureThis using custom AI/ML models to identify plant diseases, how likely would you be to invest? Would you write a million dollar check, or would you be worried your investment would be undermined by the expanding prowess of generalized software like GPT-4o?

I would not write that check. In fact, many people who have *already* written checks are finding their custom applications easily (and cheaply) replicated by generative ai.

Water level monitoring, long-form weather descriptions, and plant disease identification are just a few random examples taken from my day-to-day life. How many millions of other use cases can be reduced to an LLM call and some prompt engineering? And if not today, what about in six months, or a year?

If you are building any type of software you need to understand the abilities and limitations of generative ai and LLMs. A significant amount of future software will incorporate, in possibly very mundane and hidden ways, some integration with a type of generative ai or LLM component. New generative ai software doing unbelievable things is exciting, but generative ai will have enormous impacts on existing software as well.

The idea that *every company is a software company* might soon be *every software company is a generative ai company.*