Team: niq
Member: Niklas Suvitie

Project 06: Bittium #4_Can AI Generate Valid IP-XACT Components?

COMP.CS.530
Tampere University

25.3.2025

is an XML format that defines and describes individual, re-usable electronic circuit designs

Problem & Solution

- **What problem are you solving?**

AI needs to make valid IP-XACT components, and validate the with Kacktus2 tool

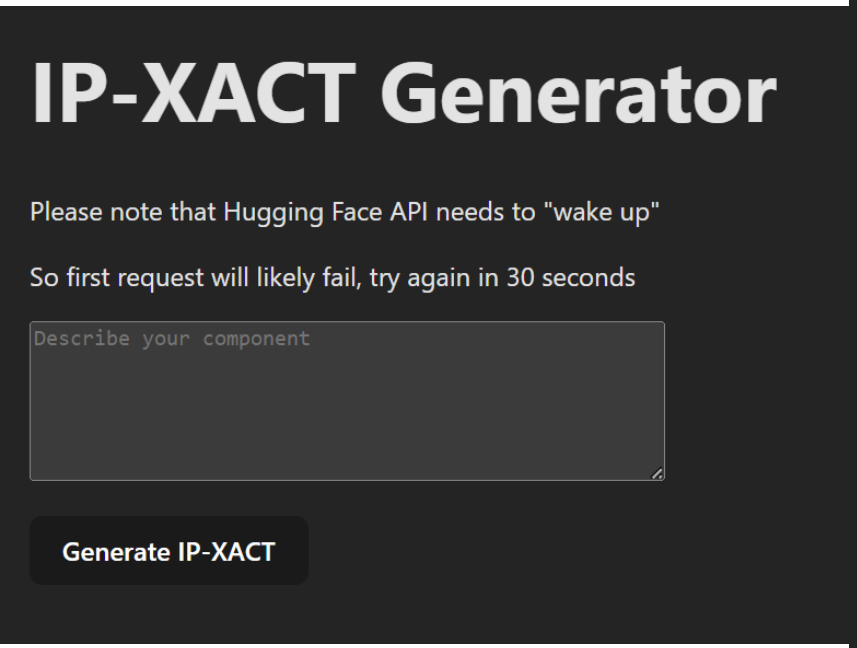
- **Why is it important?**

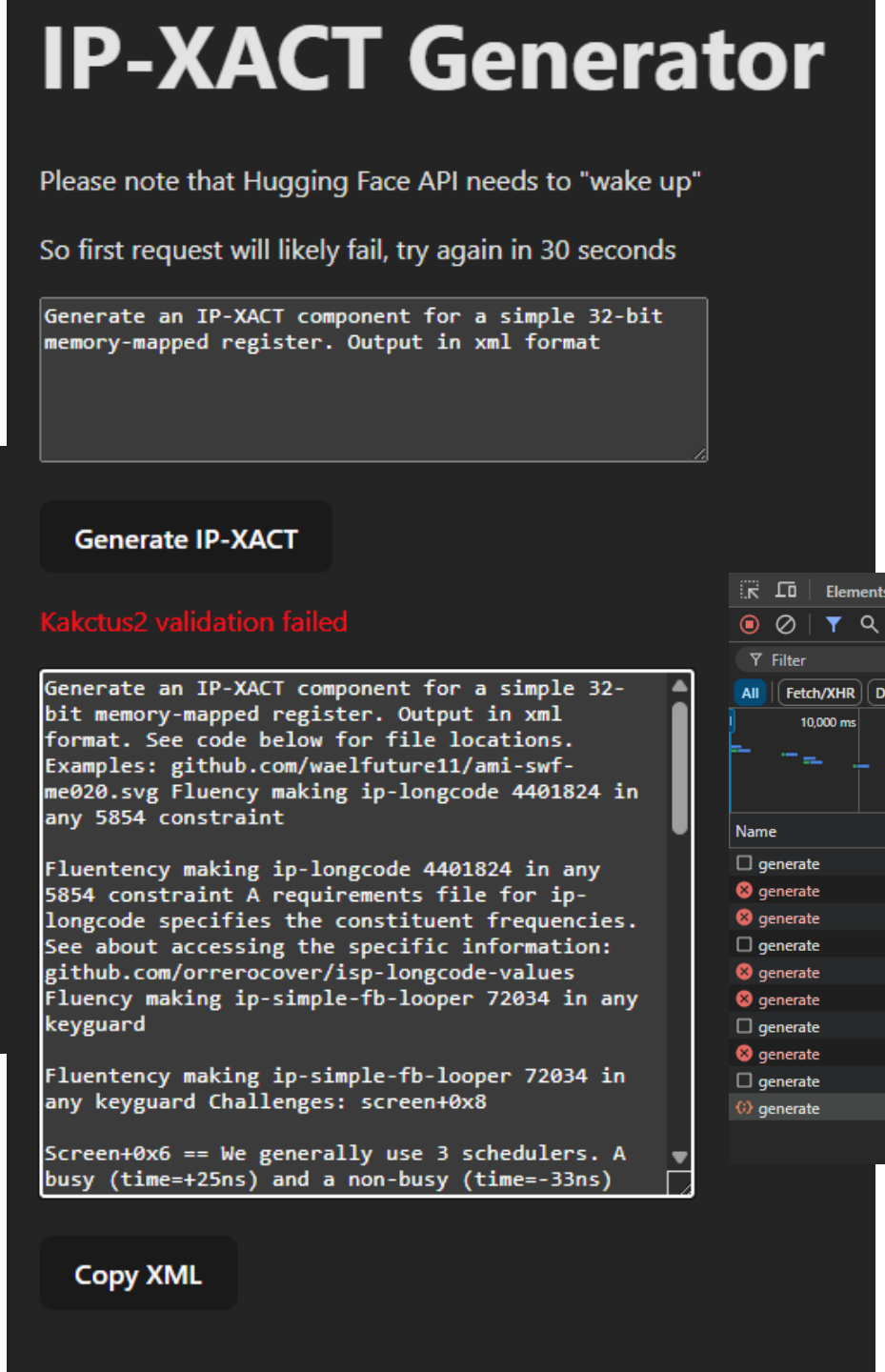
AI might hallucinate, pro grade tool ensures validity of the output

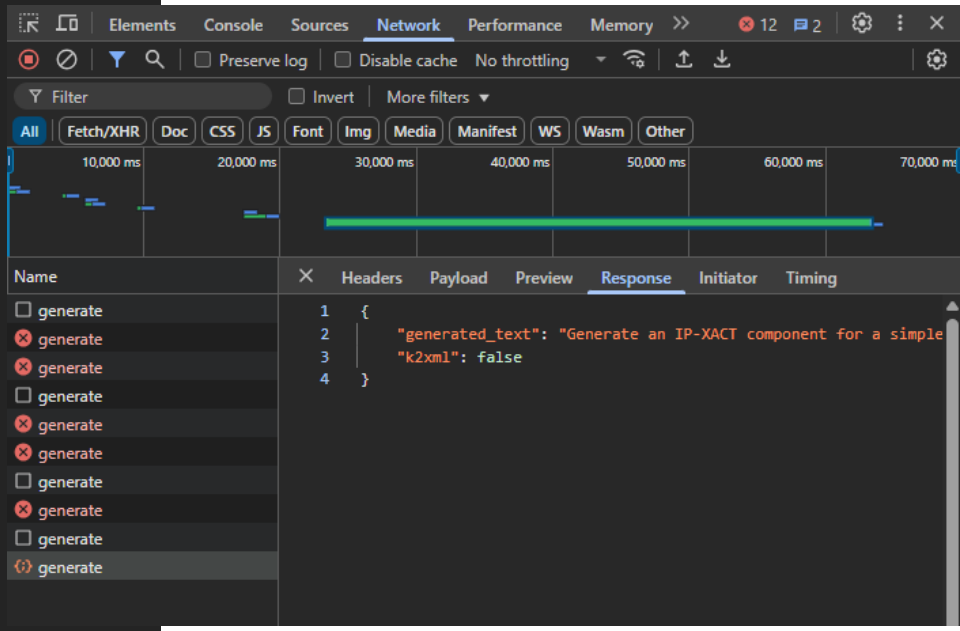
- **How does your solution address it?**

Validation tool in docker and part of the AI output pipeline

Demonstration







- Took about 45 seconds 🤯 🤯 🤯

Technical Implementation

- **Tech stack used**

React, Python (Flask), Docker, Hugging Face, Google Colab, kakctus2* & Chat GPT

- **Unique technical aspects**

Insane Docker setup: Run whole setup with 1 command

```
docker compose up --build
```

- **Key challenges & solutions**

Selecting correct model	Trial and error
Using Kakctus2 via CLI	Found Kakctus2 xml version converter, forked it, edited the code and build my own version which can be used by CLI

Impact & Future Scope

```
# Load API key from environment variable (for security)
HF_API_KEY = os.getenv("HF_API_KEY")
HF_MODEL_URL = "https://api-inference.huggingface.co/models/niklassuvitie/gpt2-medium-ipxact"
```

- **Who benefits & how?**

Anyone who needs to create valid IP-XACT, *my pipeline is free to use*

- **Real-world applications.**

IP-XACT is used in the real world :D

- **Next steps & improvements**

Better finetune, error handling,

UX improvements and bug fixes

Pretrained Model Output:
[{'generated_text': "Generate an IP-XACT component for a simple 32-bit memory-mapped register. Output in xml format.\n\nIn addition to setting up a simple input/output loop this package features:\n\nRegistering a user's account (useful for storing the user's account information in the registry)\n\nCreate a local copy of a user's account (useful for storing the user's account information in the registry)\n\nManage users with a profile (used for storing the user's public profile information to a file)\n\nPerform basic administrative access (useful for logging into the system, managing the environment, accessing files and directories in the system)\n\nPerform simple machine administration tasks, like create a local copy of the machine to access and perform initial setup of the system (user account on this server or user account or user profile on the machine, respectively)\n\nCreate/deleterate/add new accounts, delete old ones and create account objects\n\nEnable/disable basic management tasks (user account/machine account)\n\nEnable/disable basic management features (machine account/user account)\n\nDisplay a list of installed extensions and applications\n\nDisplay/list the installation status and installation progress and history of extensions in the system.\n\nCreate local version of an extension\n\nCreate account object using local file\n\nDelete a local copy of the extension and register a new extension\n\nRegister a local copy of a local extension\n\nCreate account object for a user's account\n\nPerform configuration file copy action\n\nCreate a local copy of the user's account using a configuration file\n\nCreate a local copy of the user profile using a configuration file\n\nEnumerate installed extensions\n\nCreate a local copy of a local profile using a configuration file\n\nEnumerate installed extensions\n\nDisplay installed extensions in the tree view and delete/rename/move it to the /extensions folder\n\n\nThe package also has other features that are designed to solve common problems faced by various users:\n\nSupport for local version of extensions using local file\n\nManage user account or profile using local file\n\nAccess a machine using local file\n\nDisplay the local version of the machine using local file\n\nEnable/disable basic management tasks and enable/disable basic configuration features\n\nDisplay installed extensions in the tree view and remove the previous version's extension(s) from the tree view"}]

Fine-Tuned Model Output:
[{'generated_text': 'Generate an IP-XACT component for a simple 32-bit memory-mapped register. Output in xml format:\n\n<? xml version = " 1.0 " encoding = " ioptions "? > <! DOCTYPE svg PUBLIC "-//Viruses//Tectonics//Marvin H. Luskin (MHT) & © 1999-2000 Vireo, Inc. All Rights Reserved. -//PATENTS// - // - -//TRADEMARK(s) - // - -//CONTENTS: svg: http://www.verizon.com // - - - > < svg xmlns = " http://www.verizon.com/xmlns/2.0/ " xmlns : xsi = " http://www.w3.org/2001/XMLSchema-instance " xmlns : xsi11 = " http://www.w3.org/2001/XMLHttpRequest " xmlns : xsi:schemaLocation = " http://www.verizon.com/xsi/2002/xsi " > < state > < xsd: xsdType >.NETFrameworkVersion 2.0 </ xsd: xsdType > < xmlns : xsi = " http://www.w3.org/2001/XMLSchema-instance " xsi : xsiLocation = " http://windows.microsoft.com/2004/xsd " >.NETFramework </ xml >< xmlns : xsi11 = " http://www.w3.org/2001/XMLSchema-instance " xsi : context = " @context/xsd " >.NETFramework </ xml >< xsd: state = " XACT - Internal Server-4 " xmlns : v = " http://www.xactsoft.org/components/xact.xsd " > < components xsd: state = " 0 " xmlns xsi:extension = " text/xml " > < xsd : componentName >Comodo</ xsd : componentName > </ components > </ xsd : state > < xsd: xsdType >.NETFrameworkVersion 2.0 </ xsd: xsdType > < dependency > < dependency >comodo</ dependency > </ dependency > </ xsd: xsdType > < xsd:'}]

Conclusion & Q&A

- **Key takeaways**

Great pipeline, just plug better LLM to it

- **Why your solution matters**

> The task told me to use certain tool
< it was not possible
> found a complimentary tool from same organization ->
< it was not suitable
> forked it
> made it work

Voittaja ei koskaan luovuta, luovuttaja ei koskaan voita – Juhani Tamminen

Open for questions