

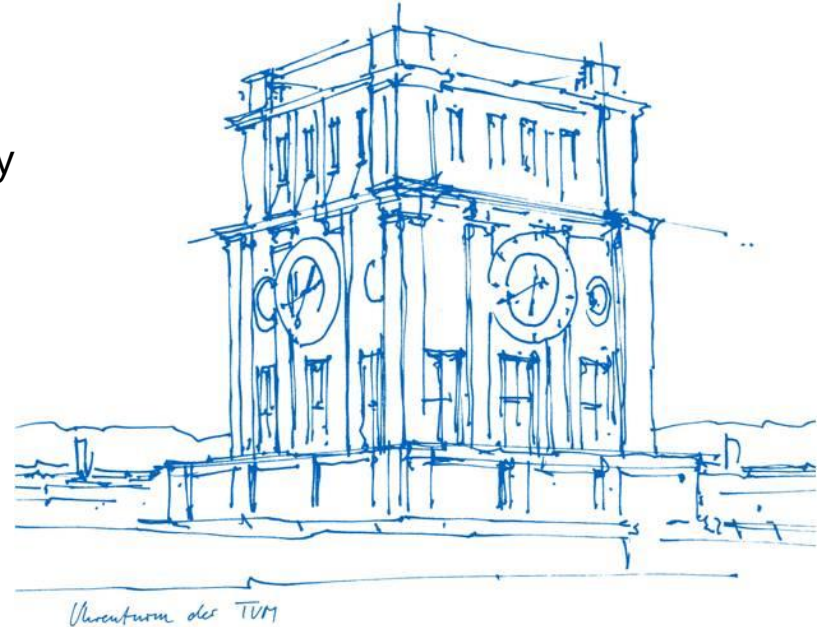
Introduction to Software Engineering

Prof. Dr. Chunyang Chen

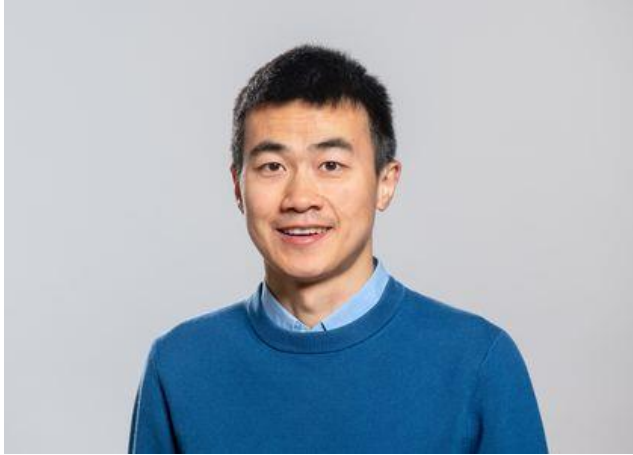
Technical University of Munich

School of Computation, Information and Technology

Chair of Software Engineering & AI



Course Team



Prof. Dr. Chunyang Chen
Lecturer
chun-yang.chen@tum.de



Yuetian Mao
Tutor
yuetian.mao@tum.de



Ludwig Felder
Tutor
ludwig.felder@tum.de

What is Software Engineering

- “The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works on real machines.” [Fritz Bauer]
- “The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software” [IEEE 1990]
- “Software engineering is that form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems.” [CMU/SEI-90-TR-003]

Challenges in Software Engineering

Uncertainty

- Volatility
- Uncertainty
- Complexity
- Ambiguity

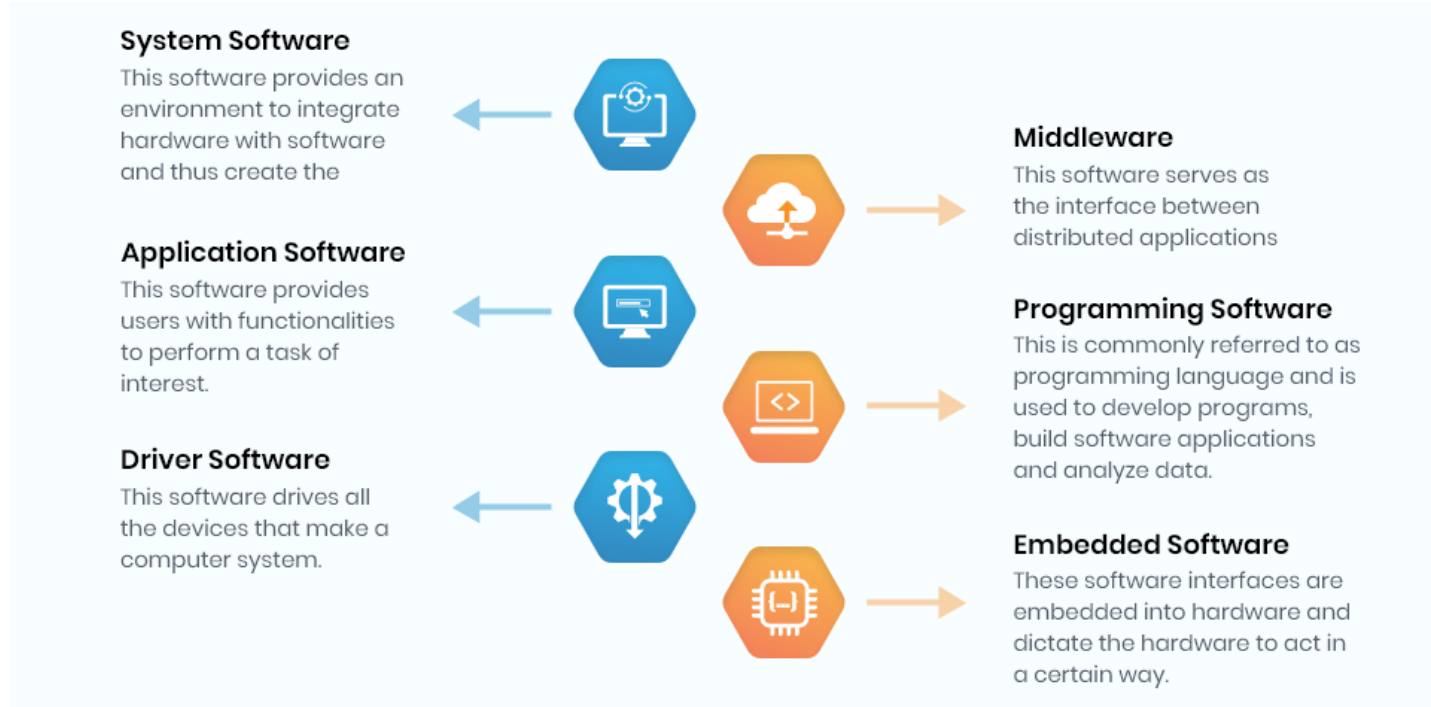
Schedule

- Delivery dates are often set before requirements are fully defined, leading to an underestimation of the needed time.

Quality

- Functionality • Safety
- Performance • Scalability
- Robustness • Compatibility
- Usability •

Different Types of Software



Different Types of Software

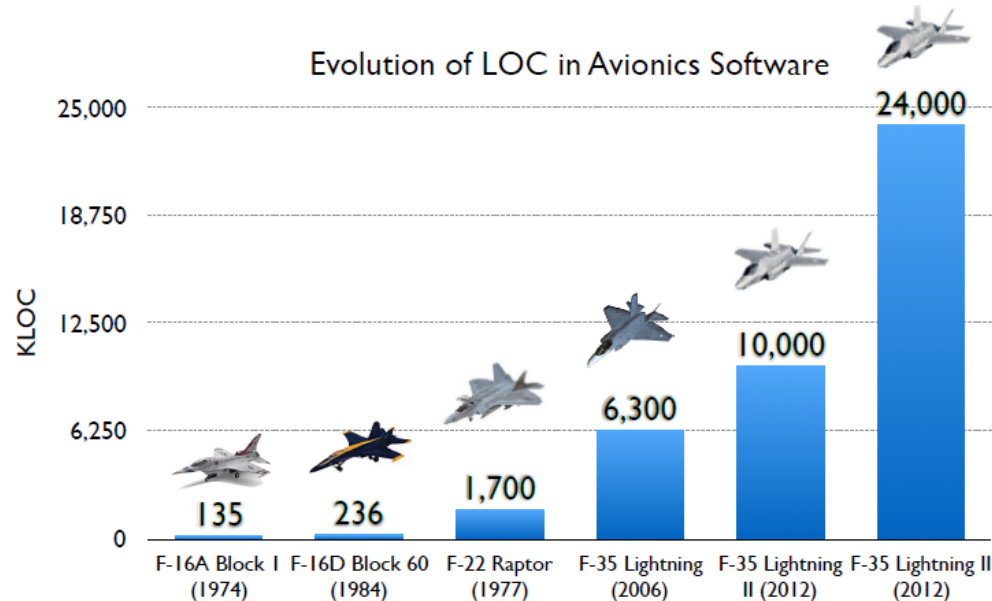


What is Software Engineering

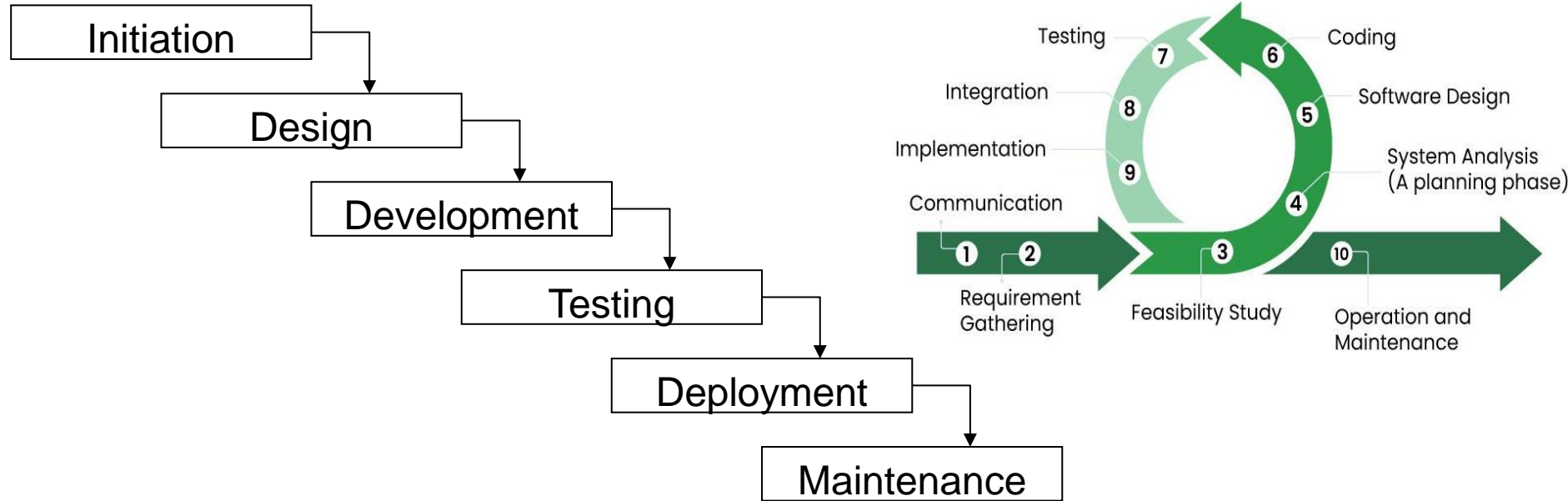
- “The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works on real machines.” [Fritz Bauer]
- “The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software” [IEEE 1990]
- “Software engineering is that form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems.” [CMU/SEI-90-TR-003]

Software projects can be BIG!

- Windows 7 had ~1000 developers working on it
- Linux Kernel had >3000 committers in 2018



Software Development Lifecycle

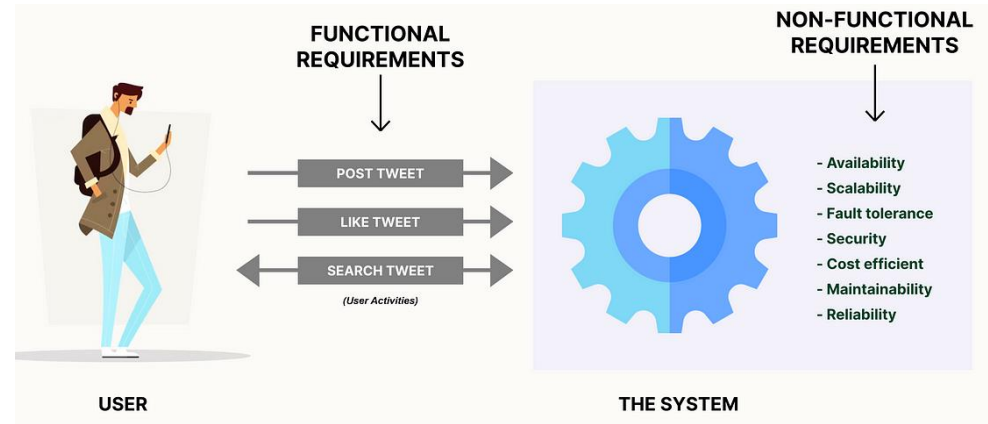


Course Overview

- Introduction to Software Engineering
- Software Requirement Engineering
- Software Architecture
- Software Testing
- DevOps
- UI/UX
- Software Security and Privacy
- Open Source Software
- AI for Software Engineering
- Software Engineering for AI

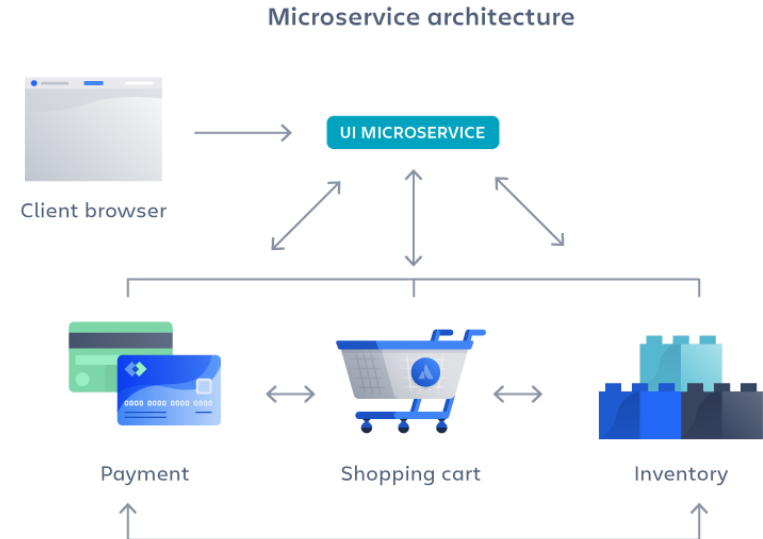
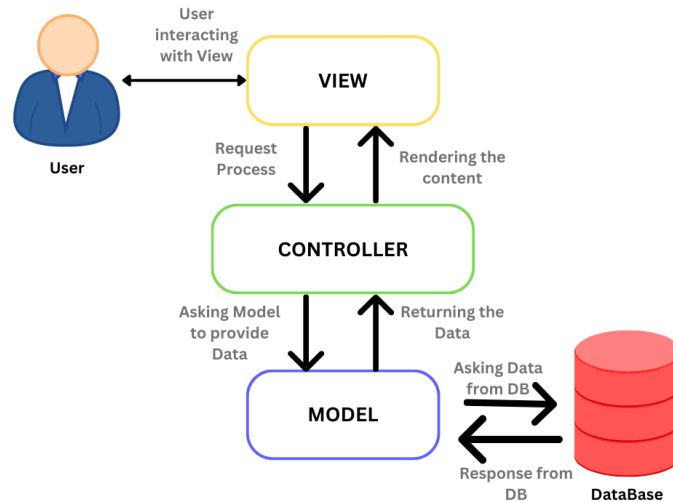
Software Requirement Engineering

- How do we determine what the software should do to meet user needs and expectations?



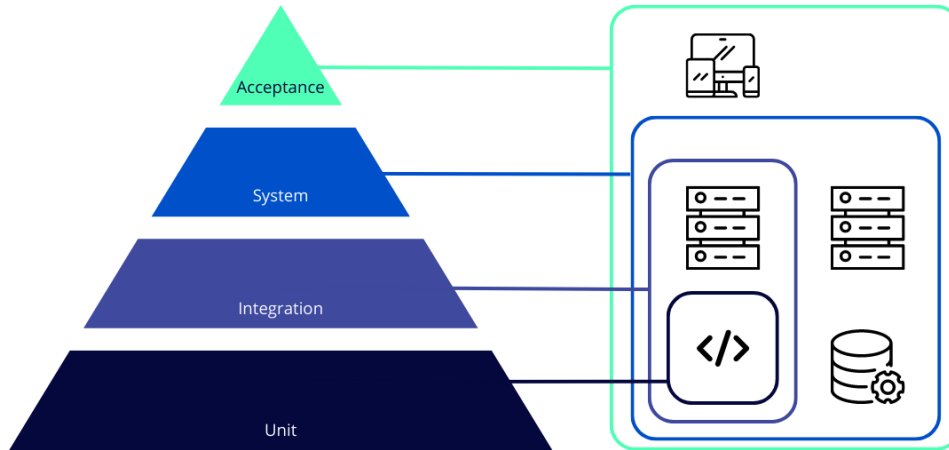
Software Architecture

- What architecture should I use for my application?



Software Testing

- How can we ensure that software works as intended and meets user expectations?



Manual Testing

vs

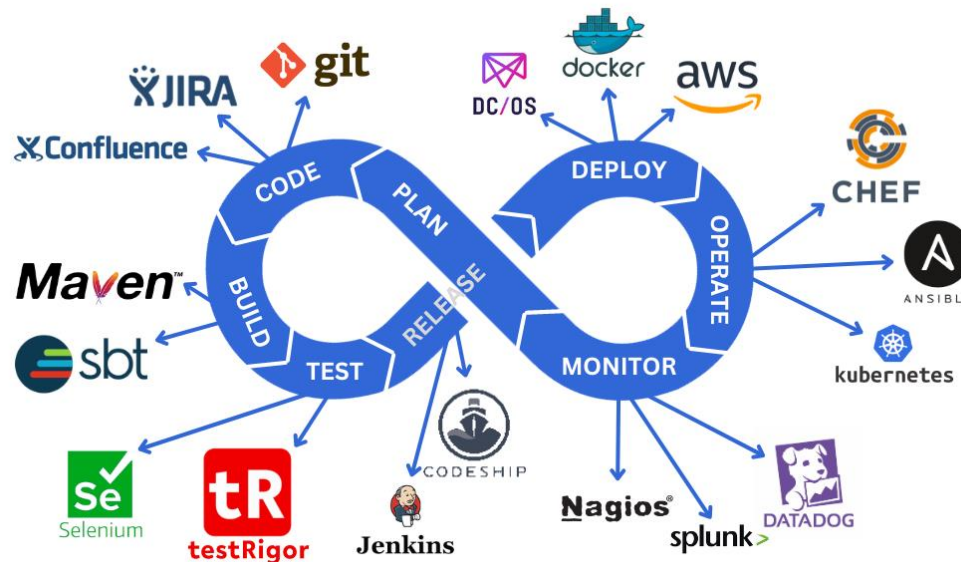
Automated Testing

- Human-executed
- Flexible & intuitive
- Good for exploratory testing
- Time-consuming
- Prone to human error
- Best for usability & UI testing

- Script-driven
- Fast & repeatable
- Ideal for regression testing
- High initial setup time
- Requires programming skills
- Best for performance & load testing

DevOps

- How can we bridge the gap between development and operations to deliver software faster and more reliably?



Functional/Non-Functional Requirements

- **What are the risks if some functional and non-functional requirements are not implemented?**

- The system must display a responsive, accessible search bar at the top of each page, allowing users to quickly find relevant content.



Users have to sift through all the products to locate what they're looking for.

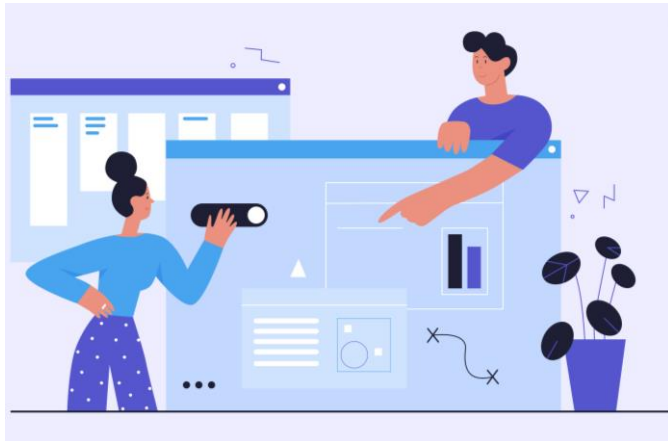
- The system must encrypt all user passwords using a secure, industry-standard algorithm (e.g., SHA-256 or bcrypt) before storing them in the database



Meta stored 600 million Facebook and Instagram passwords in plaintext, making these passwords easy to be misused by hackers

Usability & Accessibility

- How can we design user interfaces that are both easy to use and accessible to everyone?



UI



UX



Usability



Software Security and Privacy

- How can we protect software systems and user data from threats?



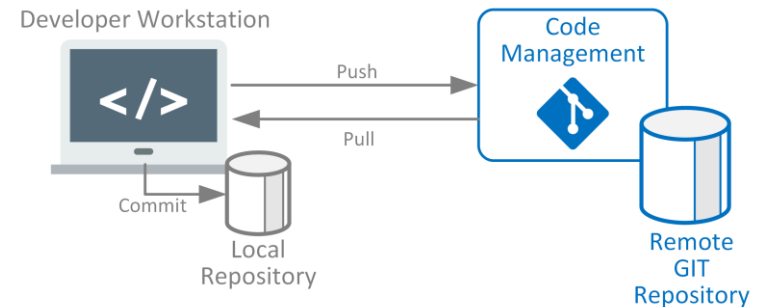
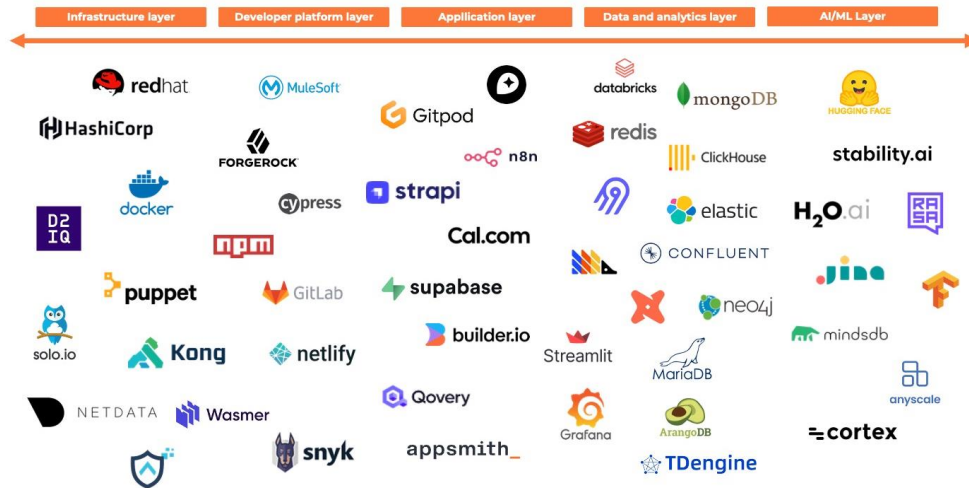
The Morris Worm (1991)



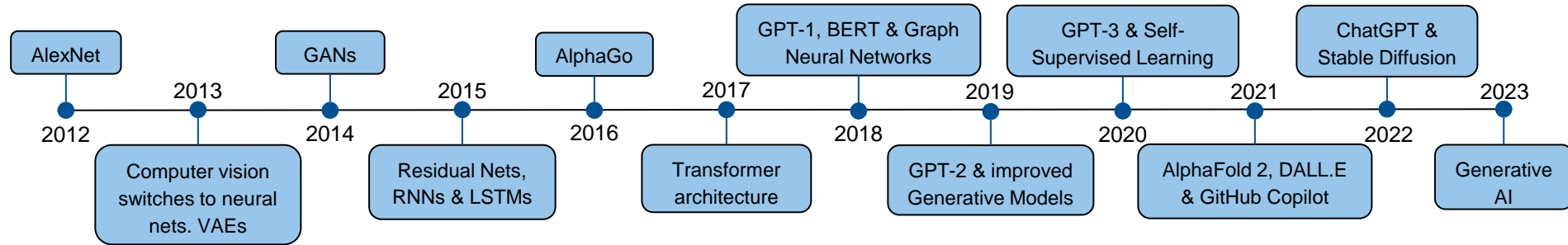
Sony PlayStation Network Attack (2011)

Open Source Software

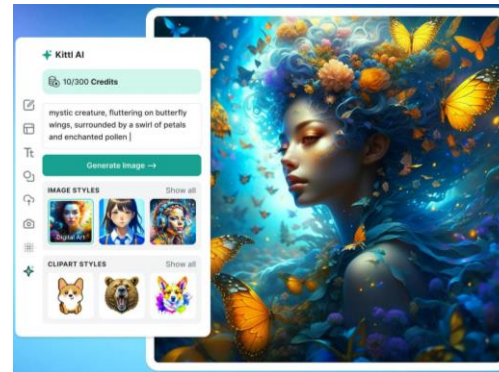
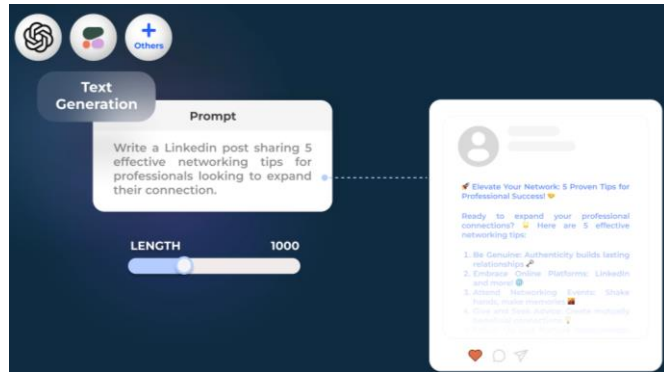
- How do open source projects ensure effective collaboration, proper use of code, policy, and adherence to community guidelines?



AI Development and Application



- **AI is evolving in Text, Image, and Video understanding and generation**



AI4SE

- How can AI/LLM tools be used to assist in software development lifecycle?

GITHUB COPILOT: CHAT

monalisa
Write a unit test for this function

GitHub Copilot

```
import unittest
import datetime

class TestParseExpenses(unittest.TestCase):
    def test_parse_expenses_with_valid_input(self):
        expenses_string = '2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK\n2023-01-03 -2.72 EUR'
        expected_output = [(datetime.datetime(2023, 1, 2), 34.01, 'USD'), (datetime.datetime(2023, 1, 3), 2.59, 'DKK'), (datetime.datetime(2023, 1, 3), 2.72, 'EUR')]
        self.assertEqual(parse_expenses(expenses_string), expected_output)

    def test_parse_expenses_with_empty_input(self):
        expenses_string = ''
        expected_output = []
        self.assertEqual(parse_expenses(expenses_string), expected_output)

if __name__ == '__main__':
    unittest.main()
```

The code assumes that the datetime module is imported. The test cases cover different scenarios such as valid input, empty input, input with comments, invalid date format, and invalid value format.

Ask a question or type '/' for commands

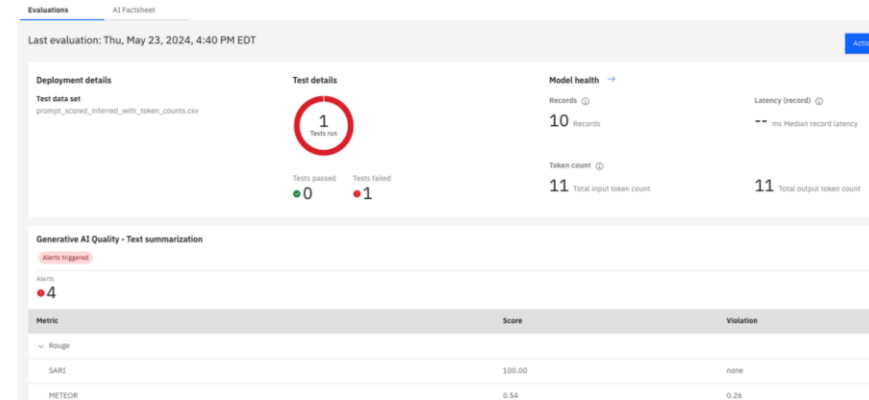
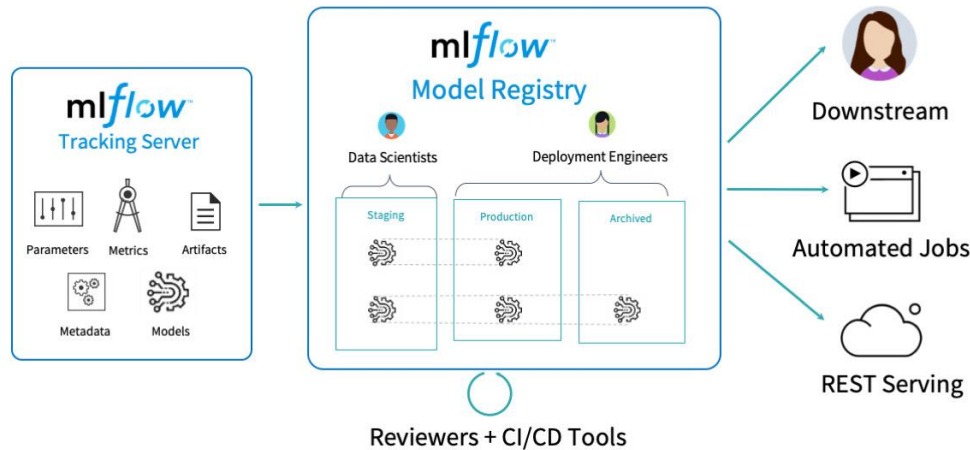
module.py

```
1 import datetime
2
3 def parse_expenses (expenses_string):
4     """Parse the list of expenses and return the list of triples (date, amount, currency)
5     Ignore lines starting with #.
6     Parse the date using datetime.
7     Example expenses_string:
8         2023-01-02 -34.01 USD
9         2023-01-03 2.59 DKK
10        2023-01-03 -2.72 EUR
11    """
12    expenses = []
13
14    for line in expenses_string.splitlines():
15        if line.startswith("#"):
16            continue
17        date, value, currency = line.split(" ")
18        expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
19                        float(value),
20                        currency))
21
22    return expenses
23
24 expenses_data = '''2023-01-02 -34.01 USD
25 2023-01-03 2.59 DKK
26 2023-01-03 -2.72 EUR'''
27
28
29
30
31
32
33
34
```



SE4AI

- What can we do to facilitate the development of AI applications?



Course Plan (Mandatory)

- Teaching schedule: **16:15 to 17:45** each **Monday** from **14.10.2024** to **03.02.2025**
except for public holidays
- **Skipped** (no lecture) next week! Tutorial is still there!
- Teaching style: Lecture + Tutorial
 - Tutorial place: **D.2.11, Seminarraum (1901.02.211)**
 - Tutorial schedule: **16:15 to 17:45** each **Tuesday** from **15.10.2024** to **04.02.2025**
except for public holidays
- Teaching resources:
 - **Moodle**

Evaluation method of this course

- **Final Paper-based Exam (100%)**
- **Team Project (Optional):**
 - Team Size: 3-4 members
 - Points: Addition to the overall score (add up to 1.0 points)
 - Peer review assessment (scaling from 0.2 to 1.2)

LLM-Assisted Application Evaluation at TUM

Students will develop an LLM-based application to assist the selection commission in evaluating master's program applications at the Technical University of Munich (TUM). This system will help streamline the Curricular Analysis process and provide recommendations based on applicants' essays and motivation letters.

Resources

- Software Requirement
 - IEEE SWEBOK V3.0, Chapter 1: Software Requirements
- Software Development Lifecycle
 - <https://medium.com/@artjoms/software-development-life-cycle-sdlc-6155dbfe3cbc>
- Software Testing
 - IEEE SWEBOK V3.0, Chapter 4: Software Testing
- GitHub Copilot
 - <https://github.com/features/copilot>