ECSE-437: Software Delivery

Majid Babaei

R3ISE
SE @ MCGILL UNIVERSITY

ABOUT ME

Name: Dr. Majid Babaei

Email: majid.babaei@mcgill.ca

Date of birth: 11 Sep. 1988

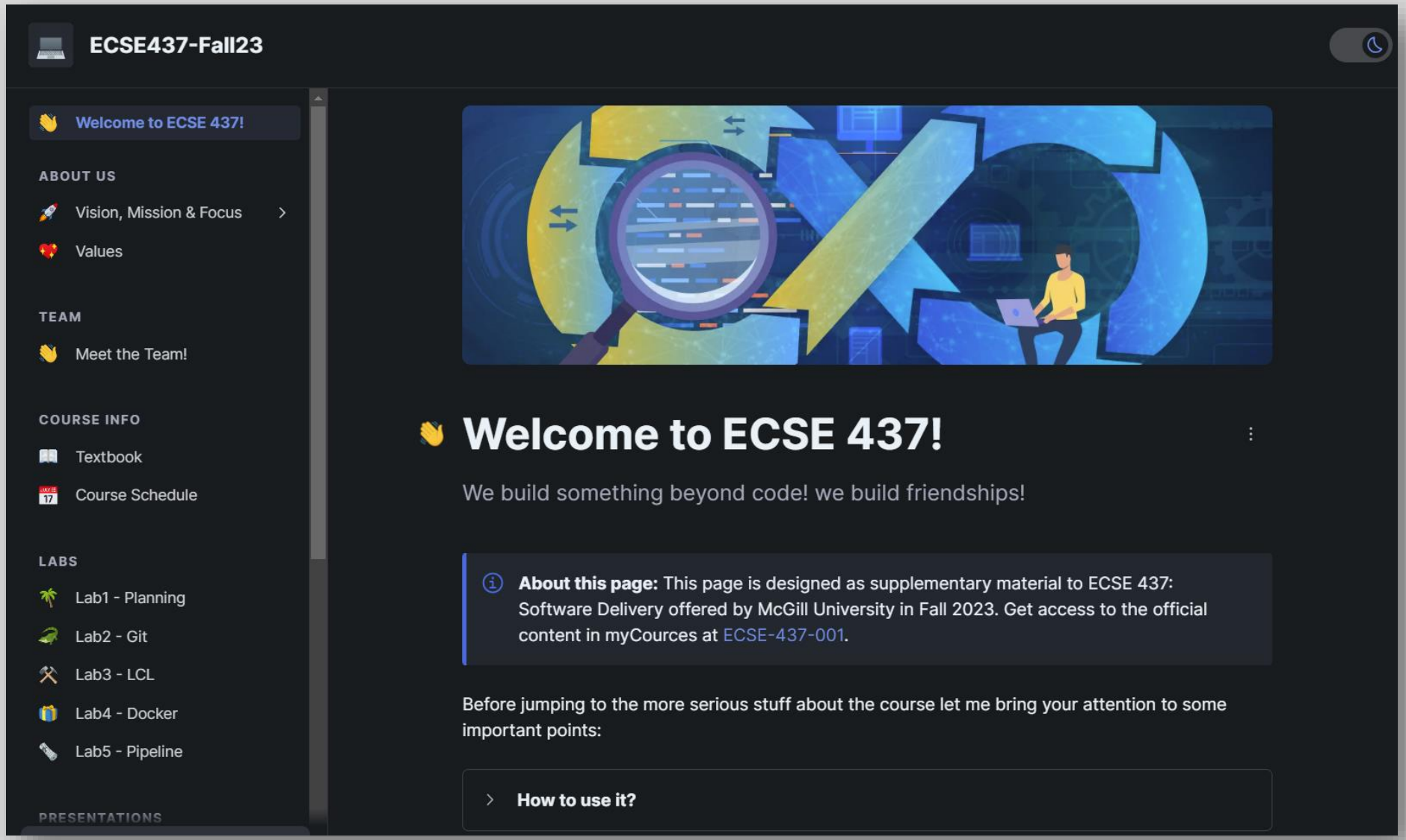
Address:
McGill University,
School of Continuing Studies,
680 Sherbrooke St W 11th Floor,
Montréal, QC, Canada H3A 2M7.

Office# 1324

SHORT BIO

No	Topics	Tools	Assessment
1	Planning	Azure DevOps	Lab1 + Quiz1
2	Version Control Systems	Git, Github, Azure Git Repo	Lab2 + Quiz1
3	Code Review	Gerrit	Quiz2
4	Containerization	Docker	Lab3 + Quiz2
5	CI/CD Pipeline	Azure DevOps	Lab4 + Quiz3

https://ecse437.majidbabaei.com



Grade Distribution

- Labs (5 labs each is worth 6%): 30%
- Quizzes (3 Quizzes each is worth 10%): 30%
- Final Project: 10%
- Final Exam: 30%
- Bones (5% presentation and 5% *excellent* learning journals): 10%

Excellent: well-structured well-explained document showing extra contributions in each lab


Textbook
Course Schedule
LABS
Lab1 - Planning
Lab2 - Git
Lab3 - LCL
Lab4 - Docker
Lab5 - Pipeline
PRESENTATIONS
Overview
Version Control Systems
Code Review
Containerization
Pipeline
Overview
Here you can find the list of all topics:
Topic 1: The Promises and Perils of Mining Git
Topic 2: Can Git Repository Visualization Support Educators?
Topic 3: Modern code review: a case study at Google
Topic 4: Who Should Review My Code?
Topic 5: Developing docker and docker-compose specifications: A developers' survey
Topic 6: Container orchestration: A survey

📅 Course Schedule

This page will be updated if any of deadlines is extended!

Dates, Topics, and assignments:

No	Date	Topics	Labs	Presentations
1	Aug. 31st	Introduction		
2	Sept. 5th	Project Planning	**Lab1** [due Sept. 19th]	
3	Sept. 7th	Git Basics 1		
4	Sept. 12th	Git Basics 2		
5	Sept. 14th	Git Basics 3		
6	Sept. 19th	Git Advanced Topics + Presentation Session 1 (2 TIME SLOTS)	**Lab2** [due Oct. 5th]	Topics 1, and 2
7	Sept. 21st	Quiz 1		
8	Sept. 26th	Code Review		

Labs

- Two tutorial sessions every week:

1:35pm -2:25pm on Mondays and Wednesdays @ RPHYS 118

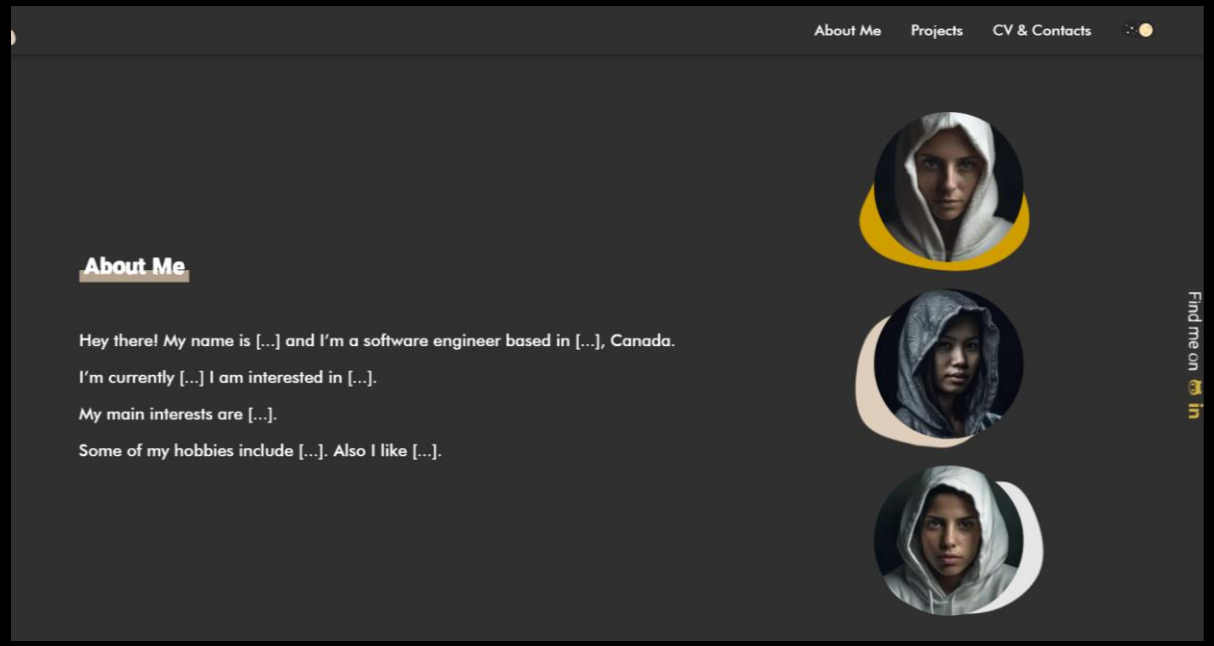
You will be assigned to a team randomly,

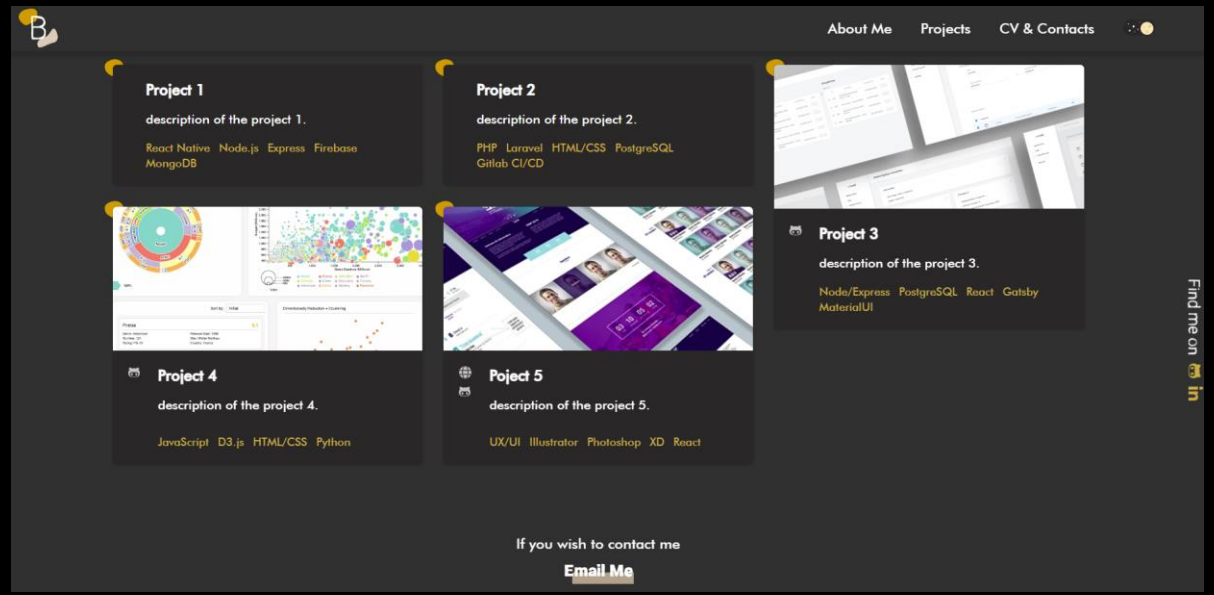
But if you have any preferences send your request to *Sarvin*

sarvin.ghiasikhalehoghli@mail.mcgill.ca

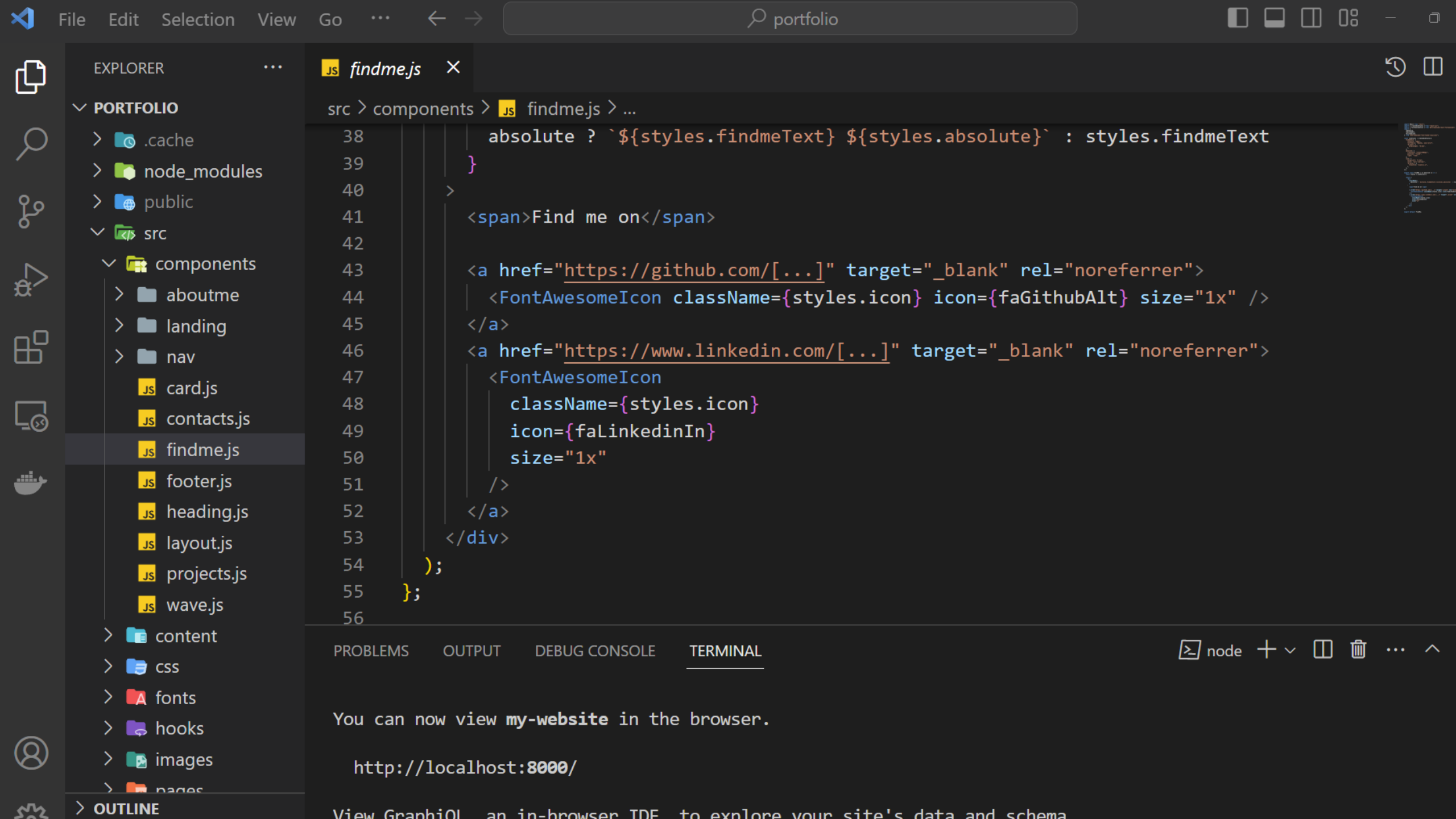
(by September 5th)

Add/Drop deadline: Tuesday, September 12
(This might impact your team!)





All faces have been generated by AI!


File Edit Selection View Go
portfolio
EXPLORER
findme.js
PORTFOLIO
.cache
node_modules
public
src
components
aboutme
landing
nav
card.js
contacts.js
findme.js
footer.js
heading.js
layout.js
projects.js
wave.js
content
css
fonts
hooks
images
OUTLINE
src > components > findme.js > ...
absolute ? `${styles.findmeText} ${styles.absolute}` : styles.findmeText
}
>
Find me on

<FontAwesomeIcon className={styles.icon} icon={faGithubAlt} size="1x" />

<FontAwesomeIcon
className={styles.icon}
icon={faLinkedinIn}
size="1x"
/>

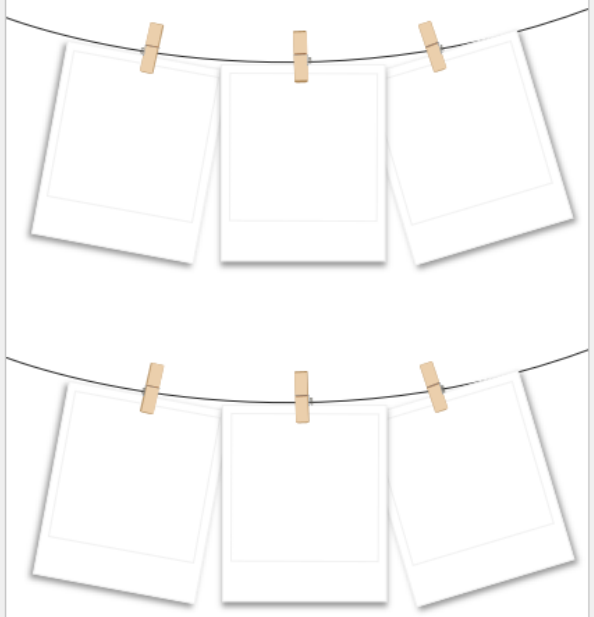
);
};
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
You can now view my-website in the browser.
http://localhost:8000/

Labs: Creating your team portfolio (*react-app*)

- Lab1: Planning
- Lab2: Version Control Systems *(git)*
- Lab3: Docker fundamentals and Linux Command Line
- Lab4: Docker advanced topics
- Lab5: Creating Pipeline

Lessons we learned.

We are at work!



Our Contributions

Questions

QUESTION #1

QUESTION #2

QUESTION #3

QUESTION #4

Description of our approach

Quizzes

- There will be 3 quizzes, each 20 minutes long

Q1: Sept. 21st , Q2: Oct. 19th , Q3: Nov. 11th

- Quizzes will be at the beginning of the class. Don't be late!

- There will be no lecture, after the quiz

Final Project

- Put together steps you have taken in each Lab and create a complete pipeline for your team portfolio react-app

- Create a 20-minute presentation video in which you explain how you connect steps and create your pipeline

- You will complete the template and add the required information to that document

TEAM PRORTFOLIO

ECSE 437
Fall'23

OUR CONTRIBUTION

Lab 1 - Planning

Lab 2 - Version Control Systems

Lab 3 & 4 - Docker

Lab 5 - Pipeline

We did more!

CHALLENGE # 1

CHALLENGE # 2

CHALLENGE # 3

CONCEPTUAL DESIGN

- **Motahareh Pour Ahimi (motahareh.pourahimi@mail.mcgill.ca)**
  - *She will mark quizzes and the final exam*

- **Sarvin Ghiasi Khalehoghli (sarvin.ghiasikhalehoghli@mail.mcgill.ca)**
  - *She will run the tutorial sessions*


QUESTIONS

Where can you find me?

9am – 3pm

7am – 9am

3pm – 7pm

7pm – 11pm

3pm – 7pm


Sharif University of Technology

Sharif University of Technology

TECHNISCHE UNIVERSITÄT
CHEMNITZ

BOSCH

Sharif University of Technology

TECHNISCHE UNIVERSITÄT CHEMNITZ

Imaginary Spaces

BOSCH

Canada

NRC·CNRC

aggregate intellect

Queen's University

McGill University

Google

Université Concordia University

What did you see?

Putting people together on the same project is not enough!

Collaboration culture must be built first!

Avoid the Collaboration Burnout

Collaborating with a decentralized workforce spread across different time zones, and inefficient use of technology are draining people.

People are busy jumping from one thing to another at the expense of having less time for deep work!

Develop a collaborative mindset is essential!

Jacob Morgan, head of The Future Organization, said, "*Collaboration can only exist in an environment where people feel safe. If you don't have an organization where people feel the ability to be vulnerable, to be empathetic, to be themselves, then you are not going to have collaboration.*"

Collaboration Must Be Purpose-Driven

Cross-team collaboration fail because it's not perceived as meaningful!

Most leaders focus on the goals and why an initiative makes sense from an organizational and business standpoint but fail to connect it with something deeper!

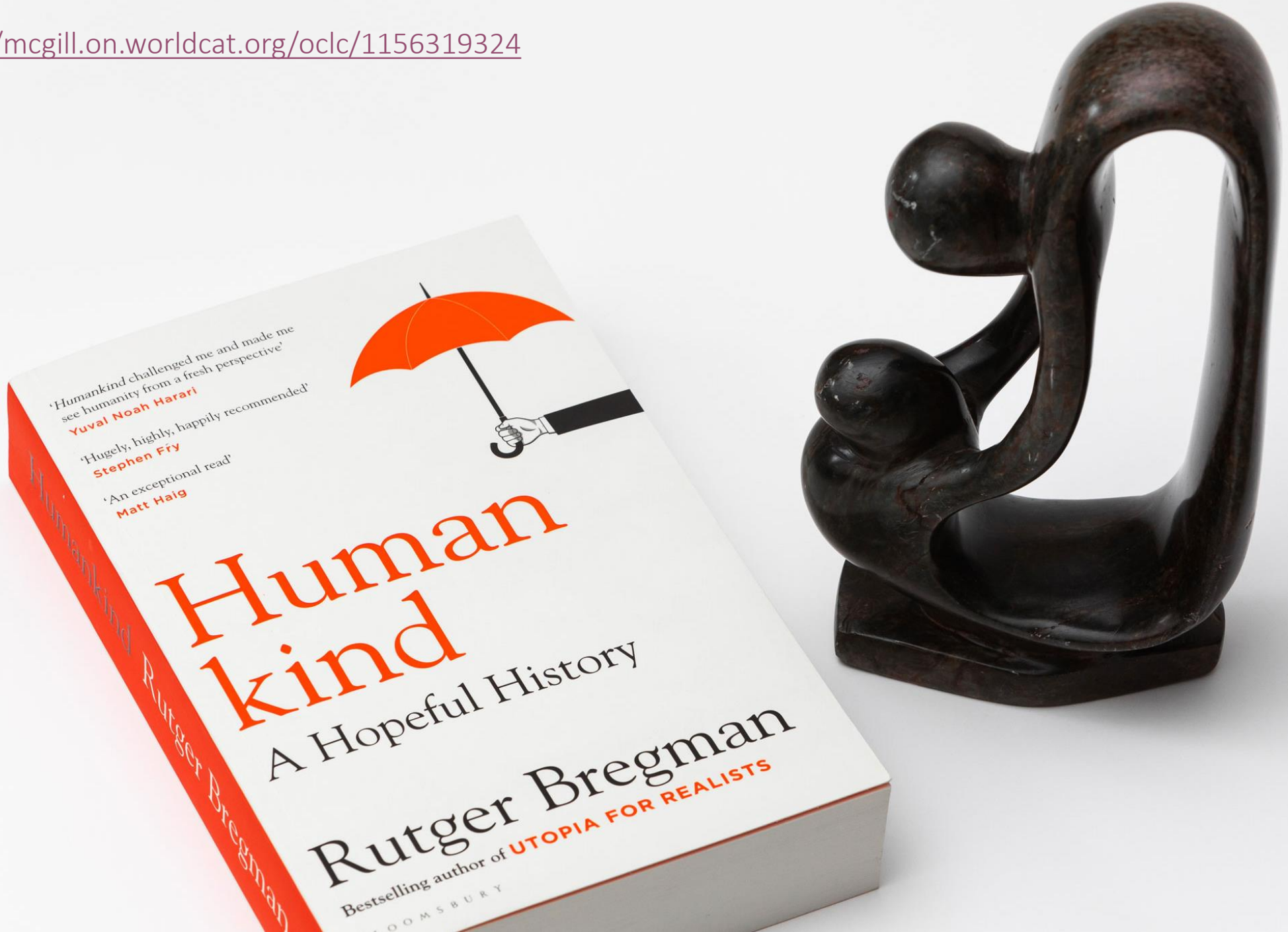
A collaboration purpose answers the *"Why are we supposed to work together?"* question.

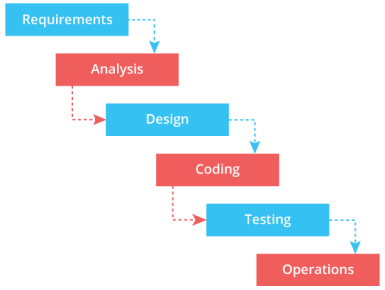
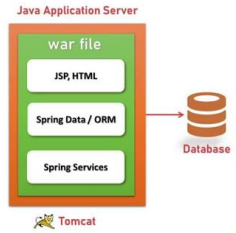
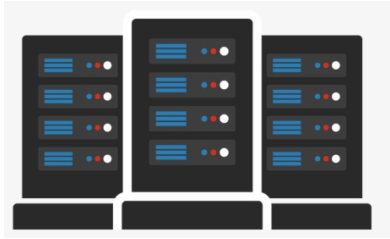
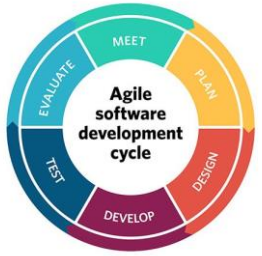
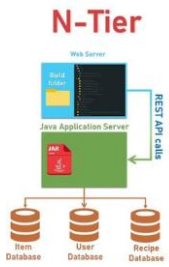
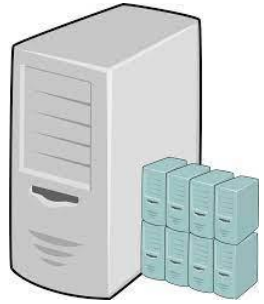
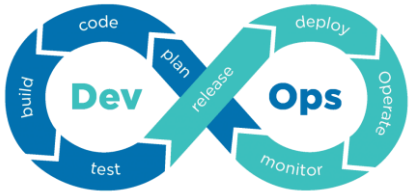
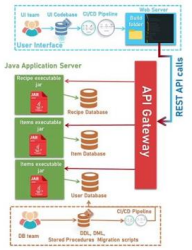
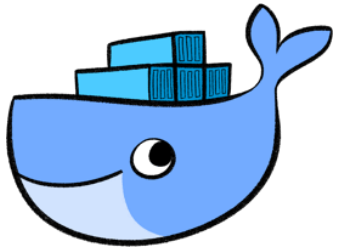
Collaboration is Not just H2H

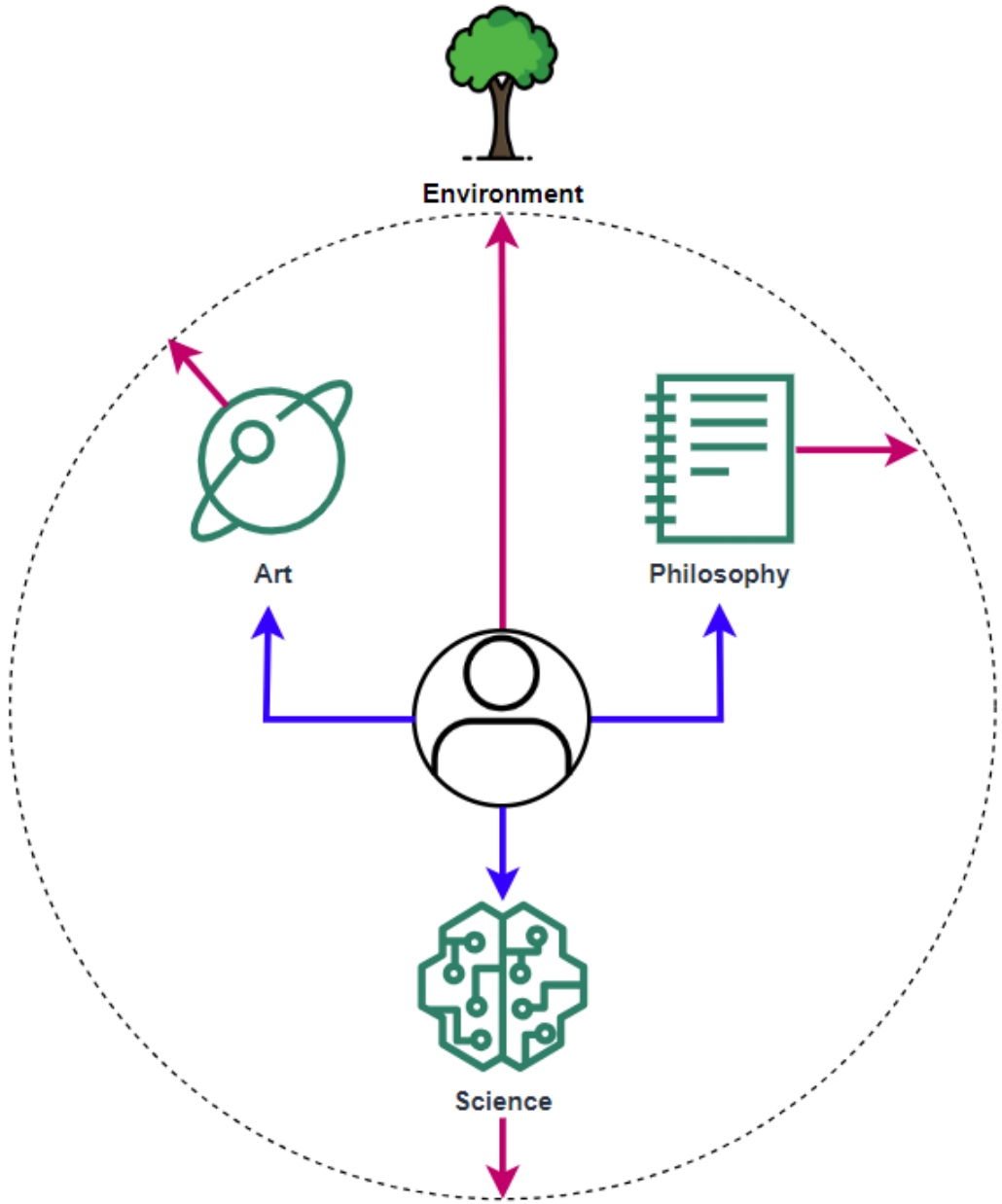
Technology will continue to play a bigger role in the future of collaboration. The rapid growth of AI will change not only *what people do* but also *shape our way of working*.

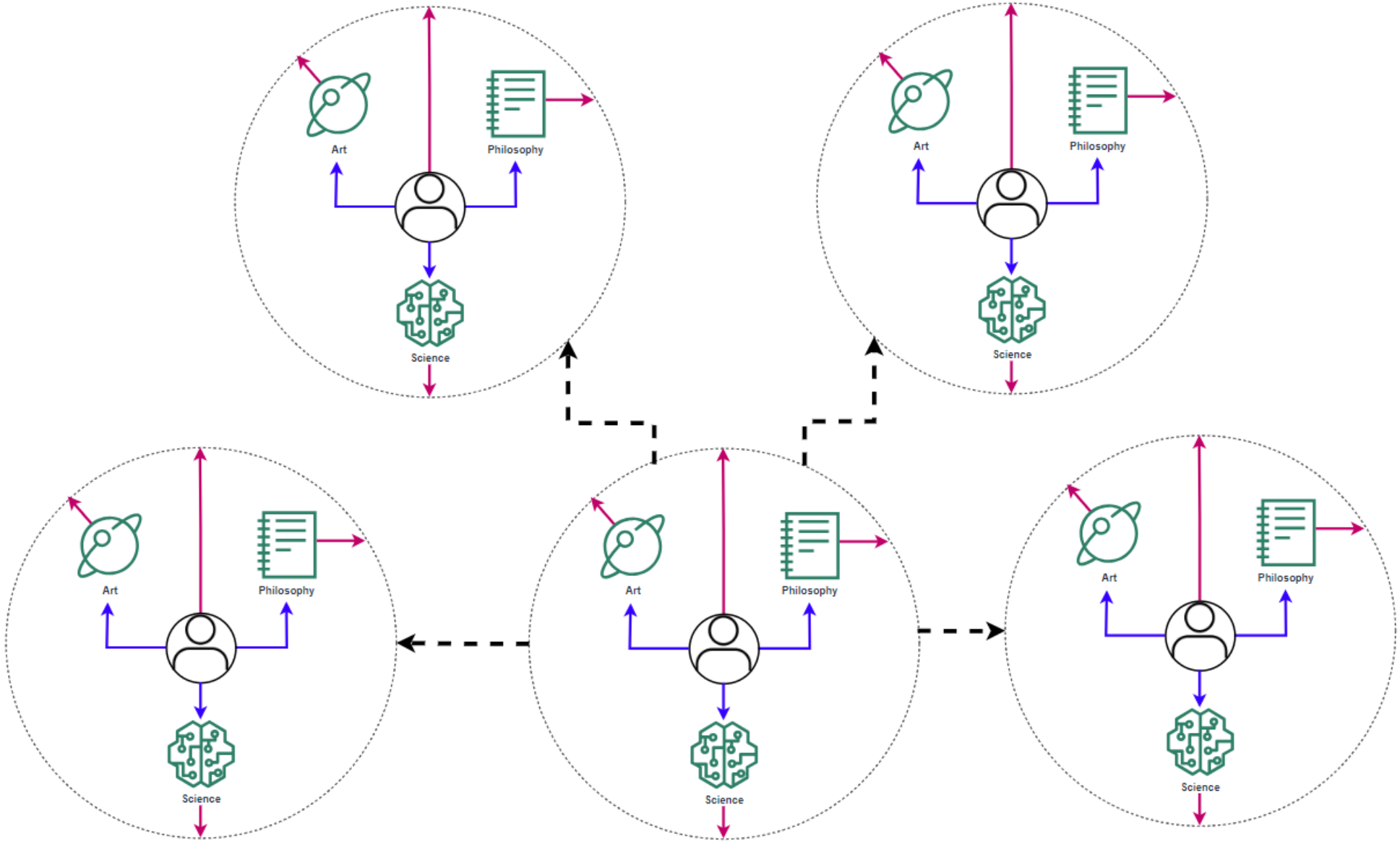
As David Coleman, author of 42 Rules for Successful Collaboration said, *"We're moving into an era where collaboration is not just human-to-human but human-to-machine collaboration and machine-to-machine collaboration."*

https://mcgill.on.worldcat.org/oclc/1156319324



Dates	Software Engineering Methodology	App Architecture	App Deployment	Data Storage
~1990 – 2000	 Waterfall		 Physical Server	 Local DCs
~2000 – 2010	 Agile		 Virtual Server	 Virtual DCs
~2010 – Now			 Containers	 Cloud DCs
Future				


Environment
Art
Philosophy
Science

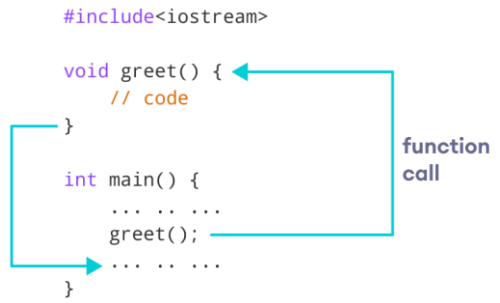

Art
Philosophy
Science
Art
Philosophy
Science
Art
Philosophy
Science
Art
Philosophy
Science
Art
Philosophy
Science

Traditional software development is dead!

Complexity is at the core!

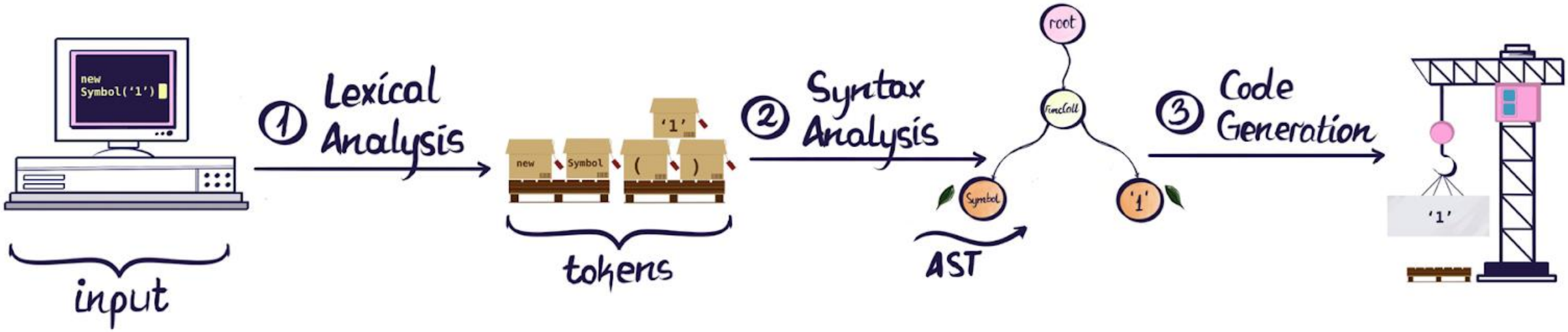
- We moved from standalone monolithic software architecture to distributed microservices.
- Increasing automation in the most repetitive and structural tasks is inevitable.
- Large Language Models (LLMs) and “Copilots” are fundamentally changing how we build software.
- Soon software engineers will no longer “write code”.

Let's break a program down!

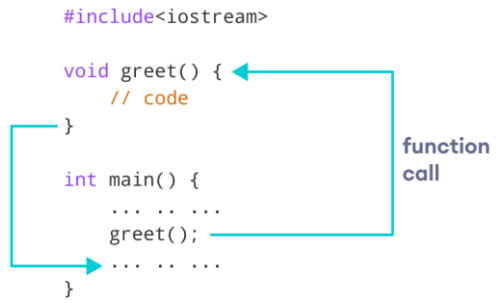


- **SYNTAX**: The formal grammar of the language, which specifies a well-formed statement that the compiler will recognize!

- In C: data_type variable_name = value_expression;



Let's break a program down!



- **LOW LEVEL SEMANTICS**: Where syntax is concerned with form, semantics is concerned with meaning.

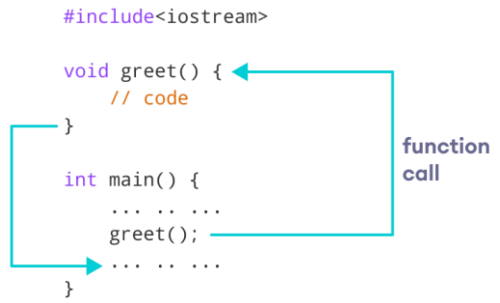
- In NLs a sentence can be syntactically correct but semantically meaningless.
  : The man bought the infinity from the store.

- At the low level: whether a statement with *correct syntax* is also consistent with the *semantic rules* as expressed by the developer using the type system of the language.

- *Whenever we have predefined rules, we can have automation!*

String firstName = 23;

Let's break a program down!



- **HIGH LEVEL SEMANTICS**: is concerned with what the code is intended to achieve - the reason that the program is being written.
- This can be expressed as pseudo-code:

```
// Check for an open trade for EURUSD
// For any open trade, close if the profit target is reached
// If there is no open trade for EURUSD, check for an entry signal
// For an entry signal, use risk settings to calculate trade size
// Submit the order.
```

- That can be interpreted by NLP models, e.g., LLMs

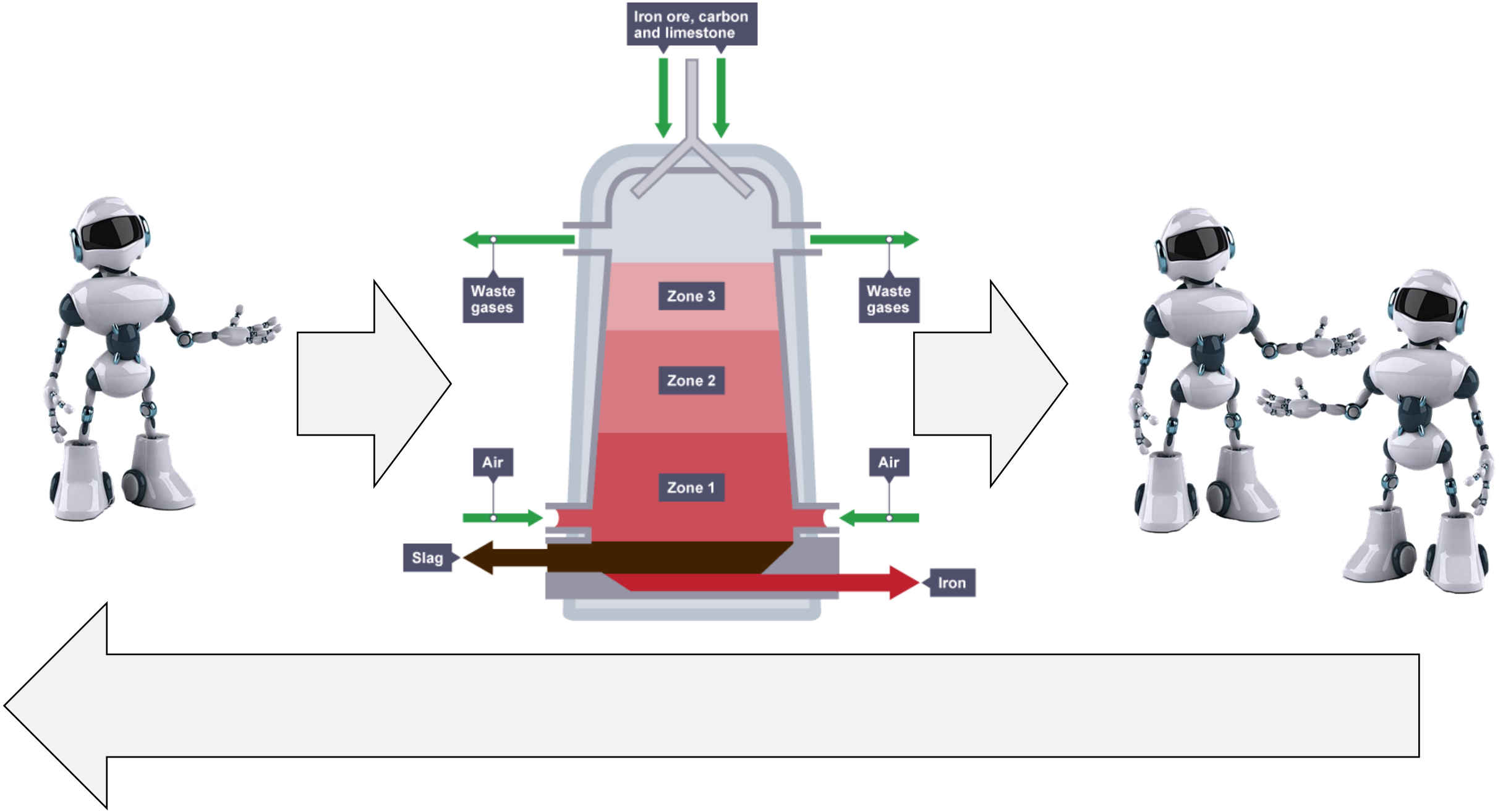
Step 1. Gather data necessary for query prompt
Step 2. Construct prompt
Step 3. Query LLM API with prompt (e.g. OpenAI's GPT-4 or Llama 2)
Step 4. Parse the results and inject them into your application

#1 INTERNATIONAL
AUTHOR OF SAPIENS

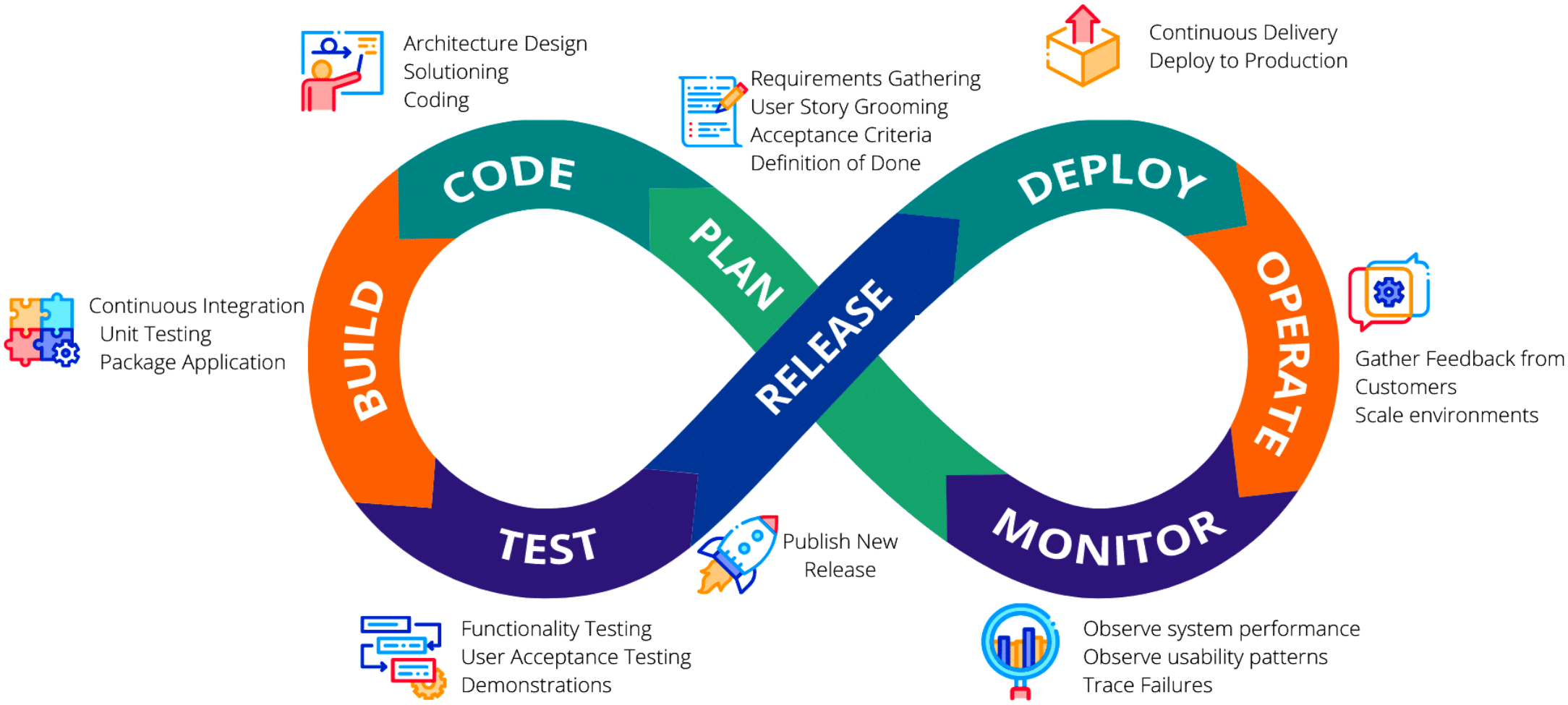
Yuval Noah Harari

21 Lessons for the 21st Century

https://mcgill.on.worldcat.org/oclc/1029771757


Iron ore, carbon and limestone
Waste gases
Waste gases
Zone 3
Zone 2
Zone 1
Air
Air
Slag
Iron

Software Delivery is impacted largely!



DevOps Culture and Practices

- A traditional approach is *dividing* people into different groups
- The term DevOps was introduced in 2007-2009 and it represents the combination of Development (*Dev*) and Operations (*Ops*)
- It advocates bringing developers and operations together within teams
- Add ***business value*** to users more quickly and be ***more competitive*** in the market
- DevOps culture is a set of practices that reduce the barriers between
  - ***Developers*** want to innovate and deliver faster
  - ***Operations*** want to guarantee the stability/quality of production systems