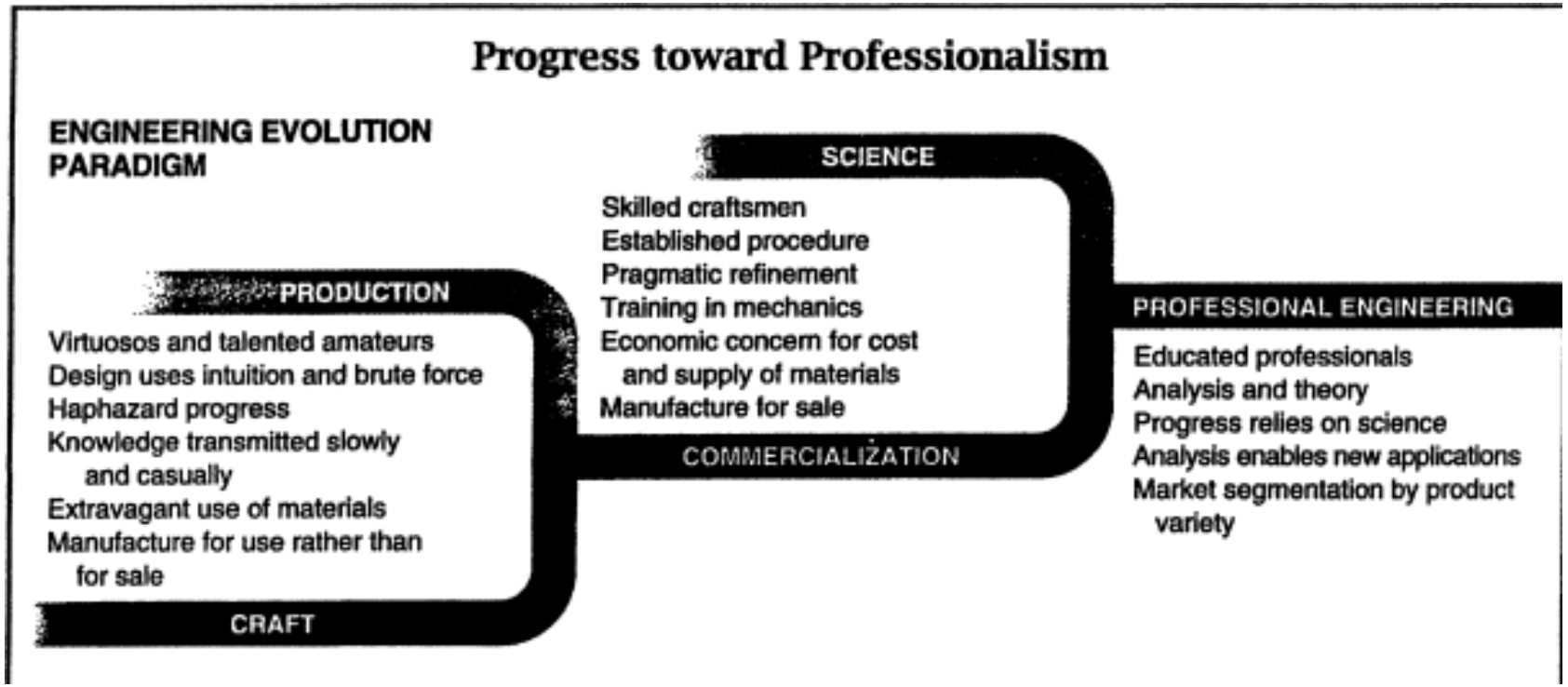
# Software Engineering

Intro

University of Roehampton London

# Why software engineering?

Understand how to apply systematic methods to the development, deployment and management of software. This is what makes this **engineering**.




https://catalogimages.wiley.com/images/db/pdf/0818676094.excerpt.pdf

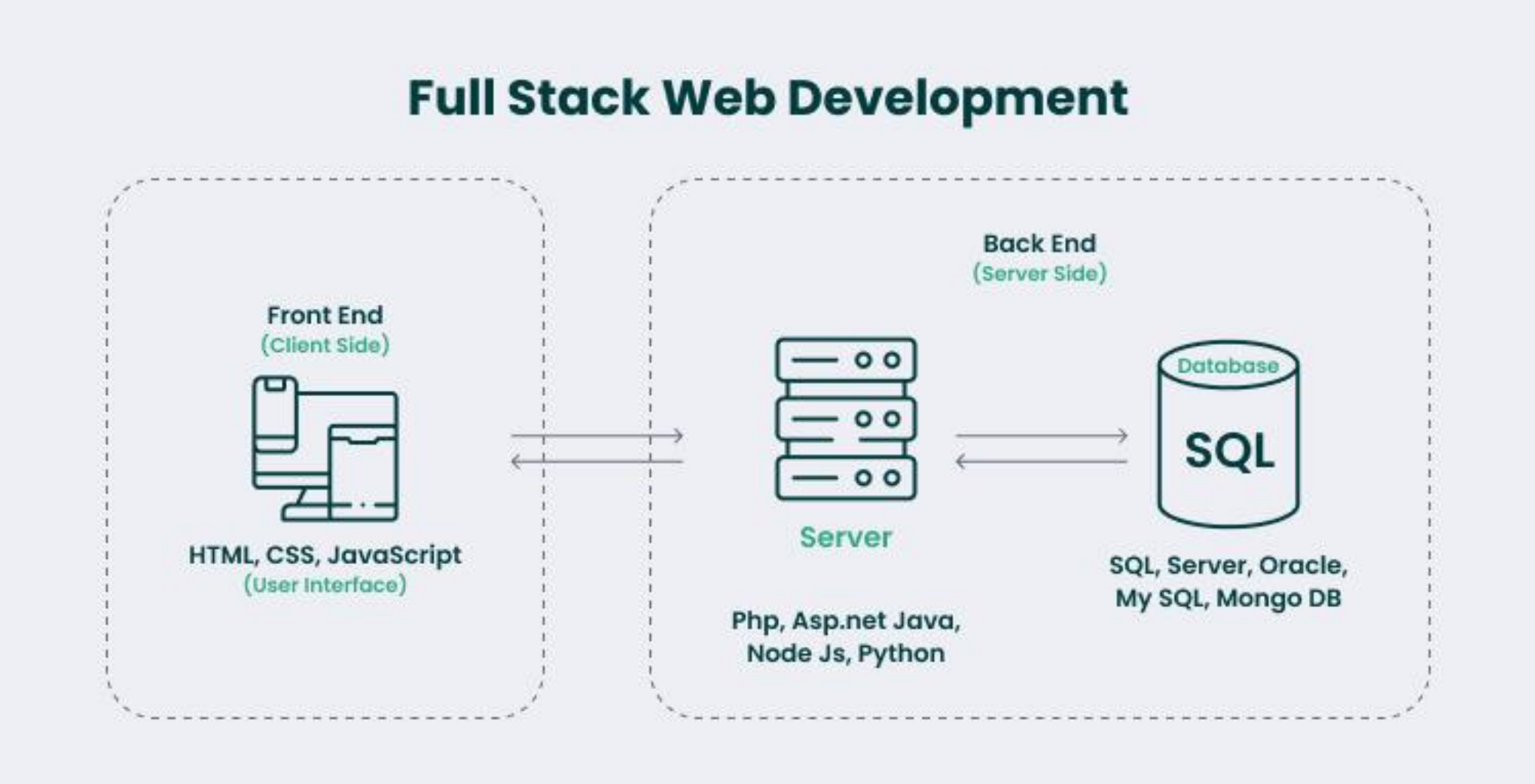
# Today

- Module introduction
- Technology stack
- Lab  1 – Development environment /HTML / developer tools
- **BREAK**
- Coursework introduction and Q&A
- Lab 2 - Git

# What happens in this module?

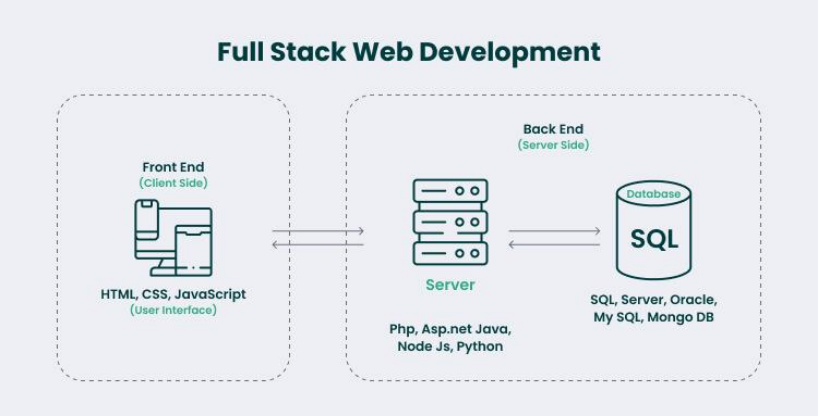
- You build a 'CRUD'/ 'full stack' web application, completely from scratch as a group project
- This is a professional practice module, we replicate standard industry practices in:
  - Technology
  - Team working
  - Planning and project management
  - Expected behaviours

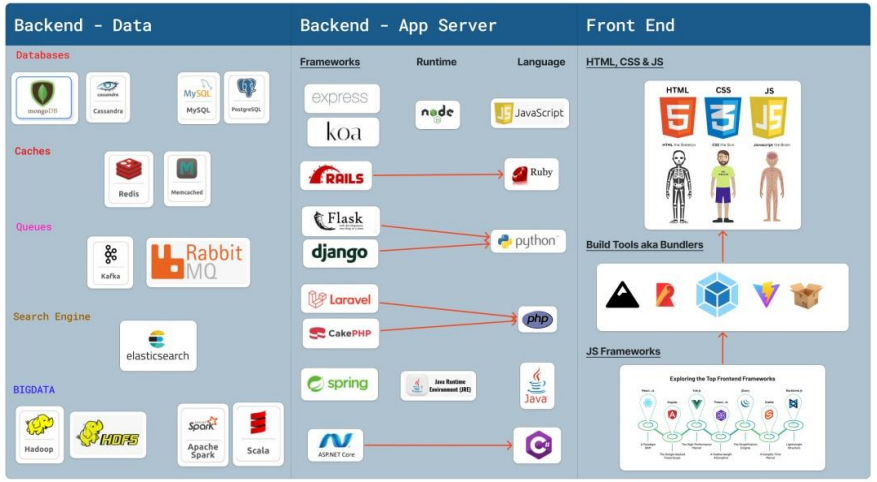
# What is a full stack/CRUD application?



# What is a full stack/CRUD application?

- Site content is mainly held in a 'backend' database (document or relational)
- Server-side 'middleware' implements business logic and prepares data for the front-end via CRUD operations* in the database
- Front-end presentation in HTML web pages or via a front-end application
- Technologies, languages, database systems, frameworks vary but the principles are the same
- This is the core architecture of web and mobile applications and will be for many years to come!




- * Create/Read/Update/Delete

# What will you learn and do?

- In the lab sessions, you are taught the skills you need and will combine these with what you already know.
- You will use these practical skills to create your group project

| Technical | Design and specification | Project management |
|---|---|---|
| HTML/CSS/JS (frontend) | Wireframes | Code of conduct |
| MySQL | Flow diagrams | Github project |
| NodeJS (backend javascript) | UML | Agile planning and reviews |
| PUG templating system | ERD and Schemas | Effective meetings, teamwork and group decision making |
| Code version control with Git | | |
| Containerisation with Docker | | |
| CI/CD with Github actions | | |

# Why?

- You will use ALL of your computing knowledge and skills
- You will also need to use your management and interpersonal skills
- This will build your 'competencies' and prepare you for work in the sector

# Why?

- You will use ALL of your computing knowledge and skills
- You will also need to use your management and interpersonal skills
- This will build your 'competencies' and prepare you for work in the sector

- It should also be fun!!

# Technology stack

What do you already know?

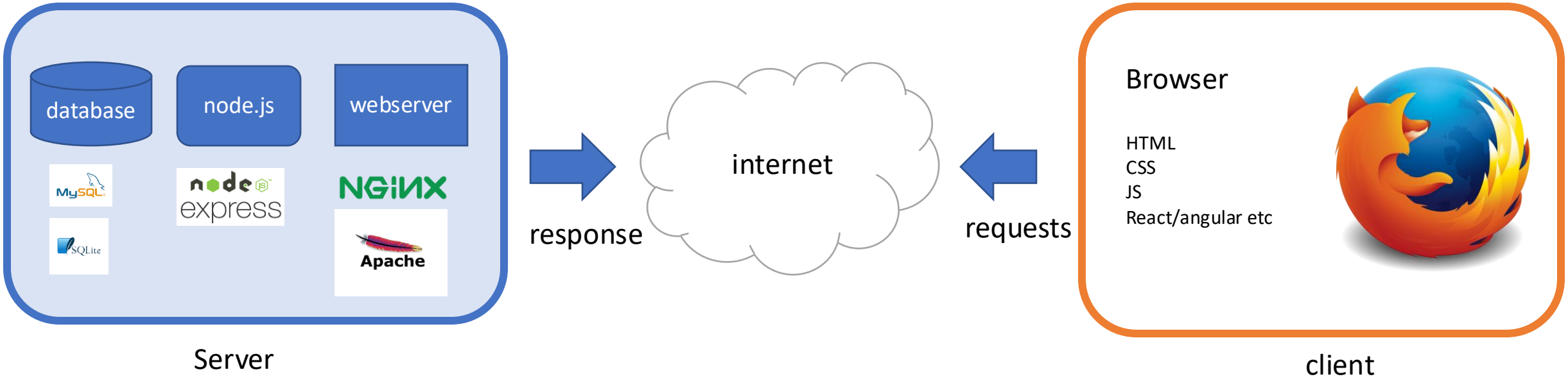
**Rankings:**

1. Never heard of it
2. Heard of it, but haven't used it
3. Basic knowledge
4. Intermediate knowledge
5. Expert knowledge

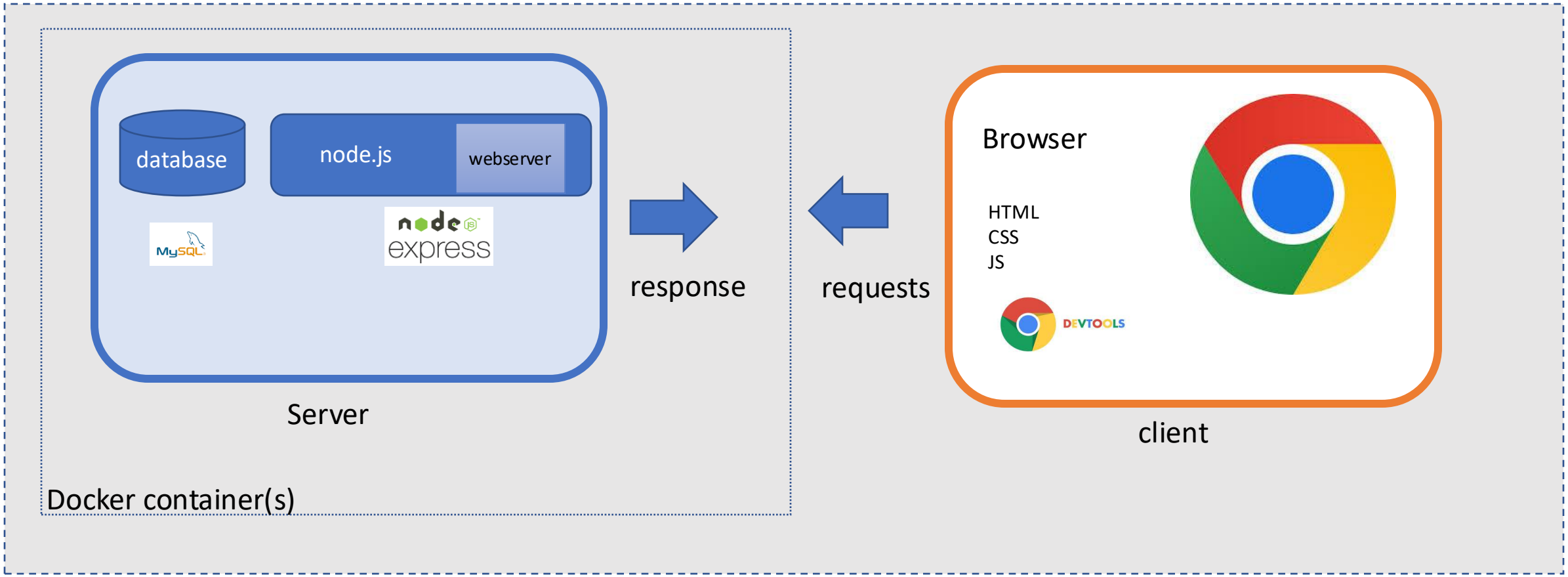
Go to menti.com and use code 2296 0025

https://www.menti.com/f2mj28bry8

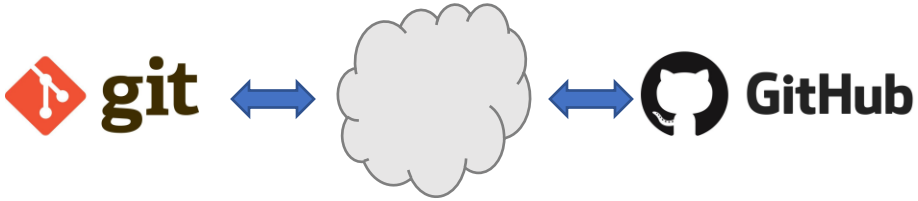
**Client/server software interaction over the internet – this diagram represents how a web application works when you access it as a user**




**Our environment**: During development, developers will have all the client and server software running on their own computer. Additionally we use tools such as VS Code and Git to write and manage source code.




Visual studio code (IDE)



# LAB1

1. Setting up VS code
2. Setting up your 'dev environment' **<- Very important**
3. Writing basic HTML and viewing in the browser
4. Using Chrome developer tools

# Break

# Group project  (Seminar sessions)

- The group project runs throughout the module and is your assessed work
- Your group will pick ONE idea.  Its OK if several groups do the same idea.
- We use the group choice module to get into your groups
- You work on the project in class time and outside
- Work is delivered in 4 'sprints'.  These are reviewed the week after the deadline and the next sprint is planned.
- You manage your work using industry standard tools
- Everyone should do a bit of everything, but group members may take specific responsibilities for certain roles.
- **You will need to do work outside of class to practice your skills and communicate with your group**

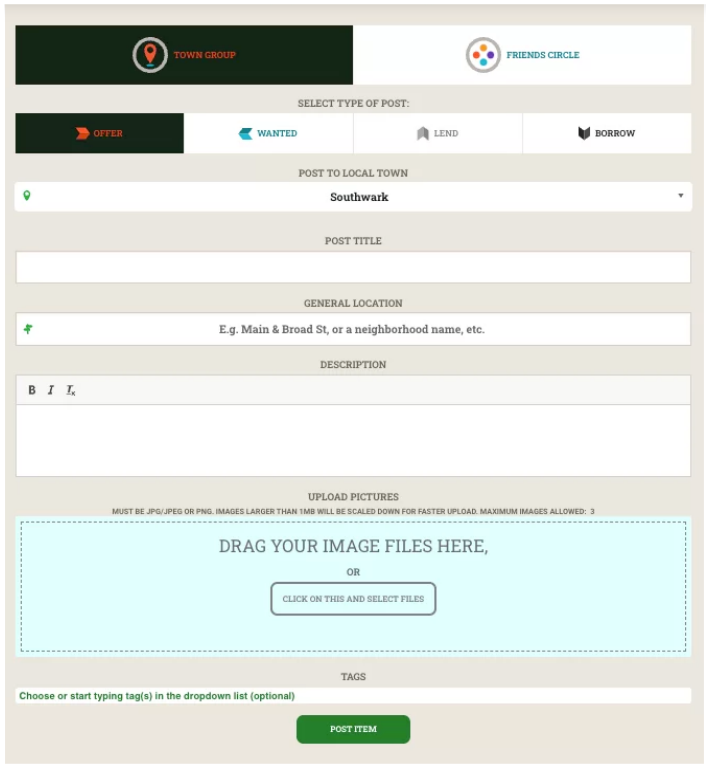
# Building your transferrable skills

- You manage your work using industry standard tools
- You will break down the work into manageable 'sprints'
- You will adopt a 'code of conduct' to help you solve problems
- I will supply you with templates to help you have effective and efficient meetings
- You are all responsible for your groups overall performance and need to contribute your best efforts

- **Evidence** is required to show you are working effectively:
  - Git (commits from all group members need to be shown)
  - Kanban boards
  - Meeting notes
  - Deliverables
  - Attendance at code reviews

# Project theme

**Sharing, exchange and building community**



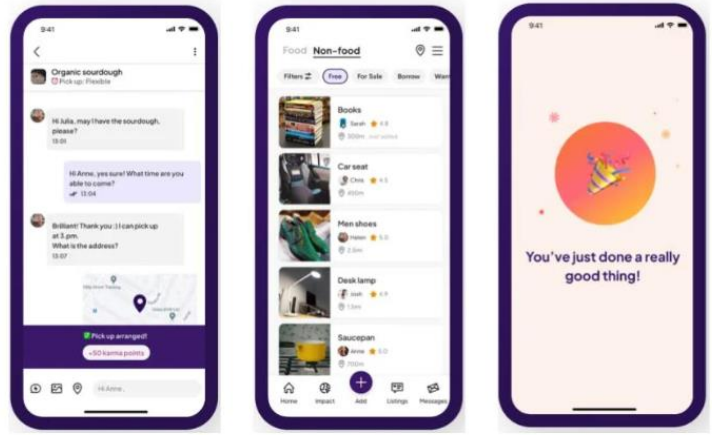


**Dress for Success Greater London** is a registered charity that empowers women into the workplace from positions of disadvantage by providing professional clothing and styling, interview coaching and ongoing support once they re-join the workplace.

An affiliate of US-based **Dress for Success Worldwide**, *Greater London* opened its doors in 2015 and carries the torch for the organisation in London.

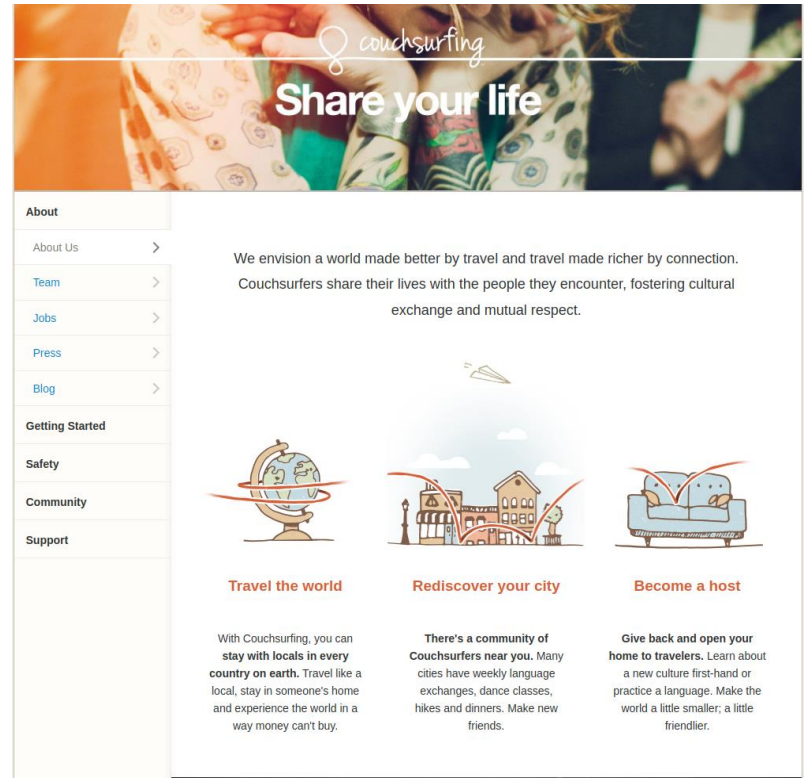
> 80% of our clients go on to secure employment.

Our Board of Trustees is committed to the organisation's success and growth. We are grateful for the on-going support of numerous corporate and individual supporters that make it possible for us to continue to help the women we serve.





freecycle.org



# Projects

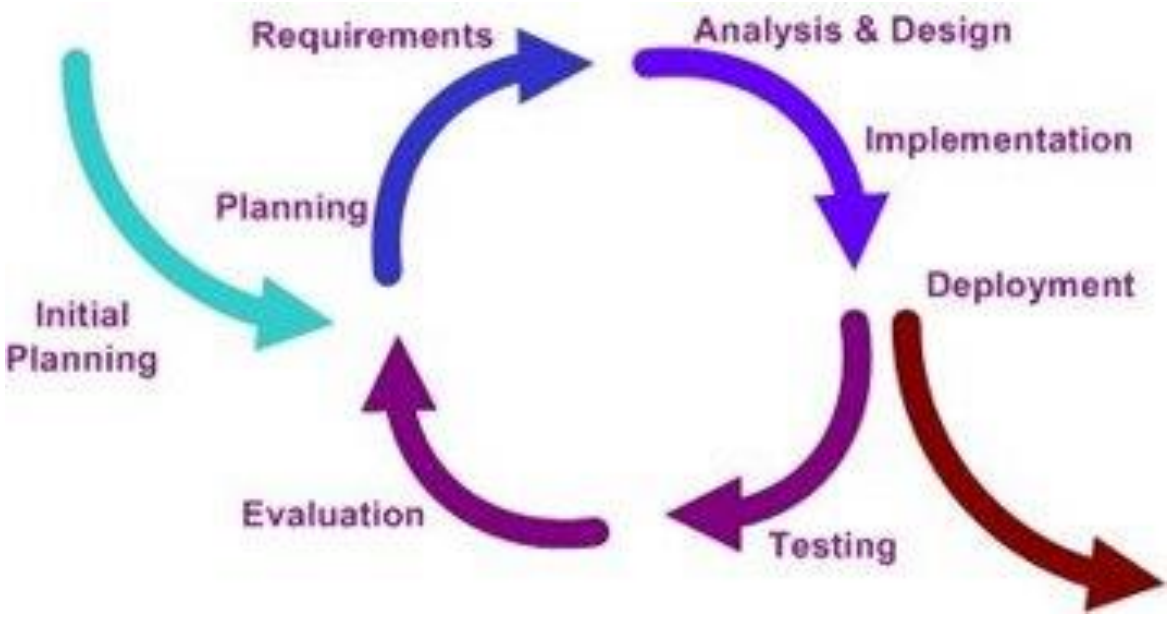
Each group will choose one of the project ideas presented by the staff team - indicative examples below. They can suggest their own idea which will need to be validated by a member of the staff team by the **end of week 2**.

Indicative project ideas

- Minimising food waste by sharing excess food
- Organising the community fridge (*this is a real project*)
- Study buddies:  peer support for students
- Ride sharing to university
- Book and record swap
- Community languages – teach and learn
- Circular fashion – re-make and mend
- Sharing outfits for job interviews, special occasions etc
- Running and cycling buddies
- Game tips and tricks

The groups will be expected to consider usability, security and ethics as they develop their application.  While all groups will start with basic functionality, the projects will allow increasing creative input and unique ideas as they develop.

**Delivery of your group project will be divided into 4 sprints, and we use the 'Agile' methodology**




**Sprint 1: Proposal and technical setup**
**Sprint 2: Designs and specifications (wireframes/UML)**
**Sprint 3: First prototype / proof of concept**
**Sprint 4: MVP + CI/CD**

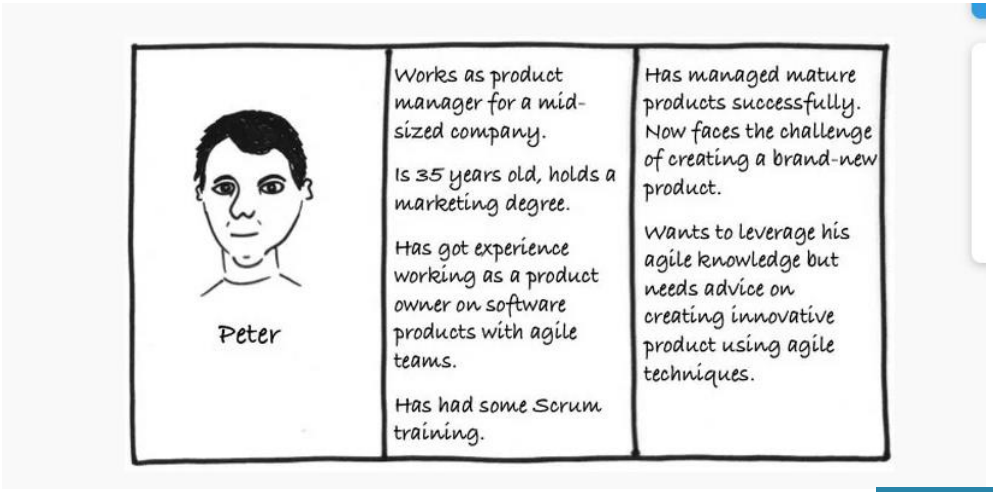
# Delivery schdule

| Time | Task | Review |
|---|---|---|
| Week 2 | Get into groups, set up project space, agree code of conduct | |
| End week 3 | Finalise project proposal (Sprint 1) | *Review week 4* |
| End week 5 | Finalise specifications (Sprint 2) | *Review week 6 (employability week)* |
| End week 9 | Complete sprint 3<br>Basic Read features | *Review week 10 – just before break* |
| Week 13 & 14 | Complete sprint 4<br>Complete CRUD features | *Presentations in weeks 13 and 14* |

*There is also an individual assignment due in week 15*

# Things you already know - Developing a specification

- Personas
- User stories
- User flow diagrams
- Wireframes
- Use case and other UML diagrams

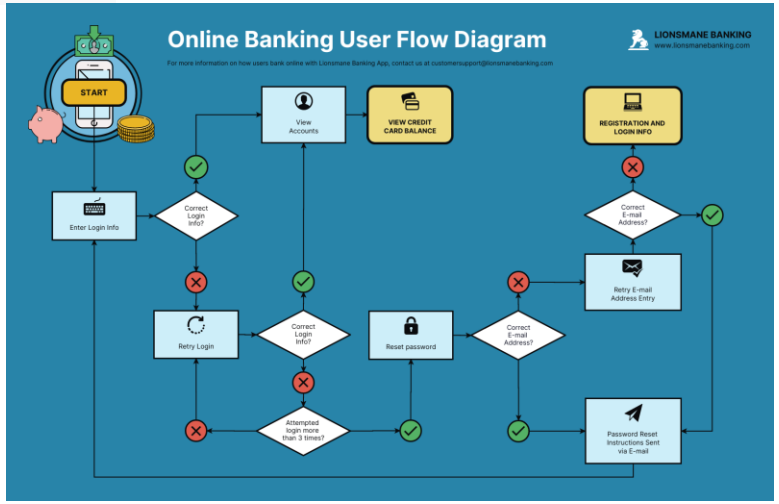




As an Account Manager
I want a sales report of my account to be sent to my inbox daily
So that I can monitor the sales progress of my customer portfolio

**Acceptance criteria:**

1. The report is sent daily to my inbox
2. The report contains the following sales details: …
3. The report is in csv format.



# Generic requirements

| Level | Requirements |
| --- | --- |
| Basic/ minimal | ☐ Users list page<br>☐ User profile page<br>☐ Listing page<br>☐ Detail page<br>☐ Tags/categories |
| Intermediate | ☐ User login<br>☐ Basic matching algorithm<br>☐ Basic user points or ratings |
| Advanced (examples) | ☐ Advanced ratings or points system<br>☐ In-app messaging<br>☐ Advanced matching algorithm / recommendations<br>☐ Use of external APIs where relevant (eg. Transport, maps, weather) |
| PM/DevOps/CICD | ☐ Github project used effectively<br>☐ Application runs in Docker containers<br>☐ At least one github action implemented |

# Sprint reviews

- Reviews take place in timetabled class time
- All students are expected to be present
- If no group members are present the mark is 0
- If group members are not present and have not give a good reason to their group the mark may be reduced

# Questions about the assignment please!!

Come next week ready to select your group and start discussions on your group project

# Git version control

# What is version control and why is it important?

- Software applications that allow us to keep a history of all source code changes – **when** changes were made and **who** made them.
- Tools for functionality required by developers such as 'branching' and 'merging'
- Git is the most commonly used – is it 'distributed'.  Others:
  - CVS
  - SVN
  - Mercurial
- Github is a commonly used platform to store and share repositories.  It is not the only way to do this with git! There are other platforms and you can self-host

# Clone a (remote) repository eg. github

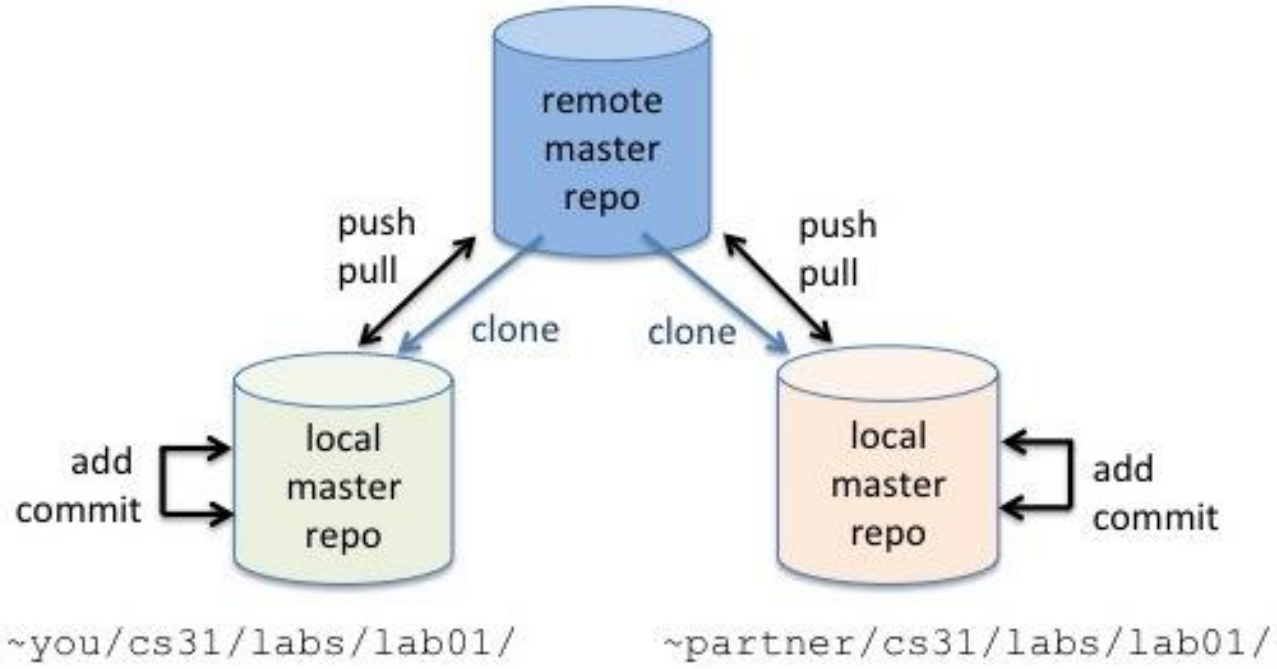

Visual studio code (IDE)

git

GitHub

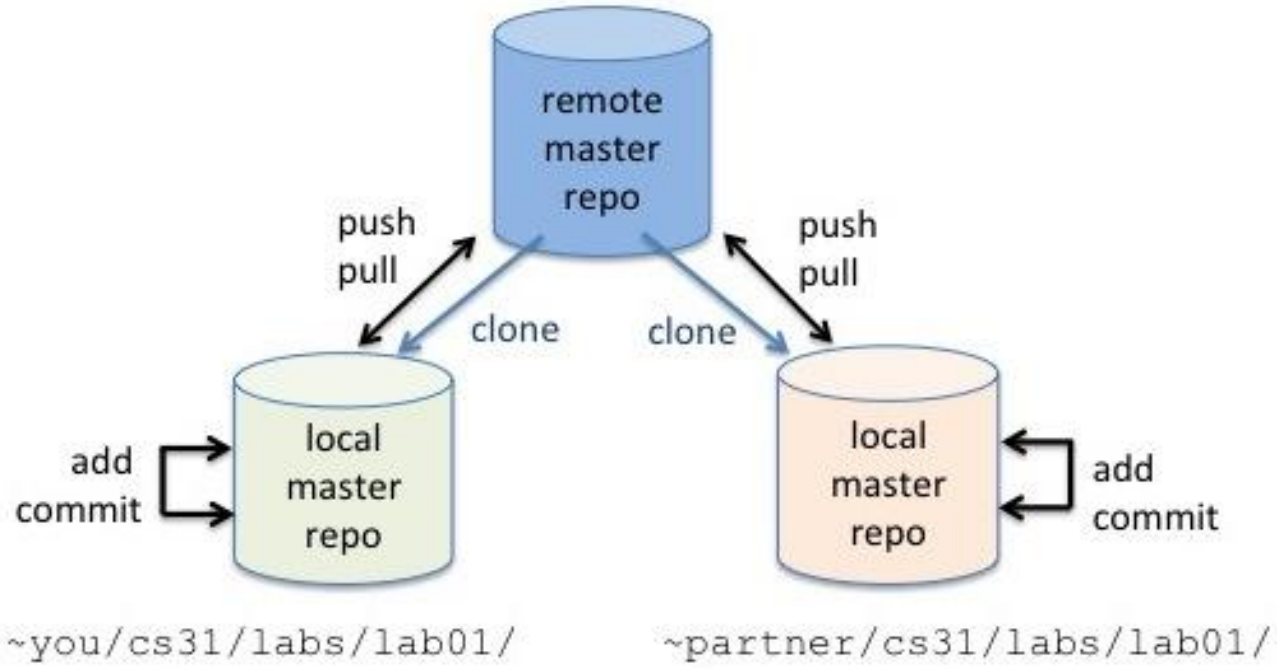
# Key concepts

- Repository
- Remote
- Clone
- Working directory
- Branch / checkout
- Commit
- Push
- Pull

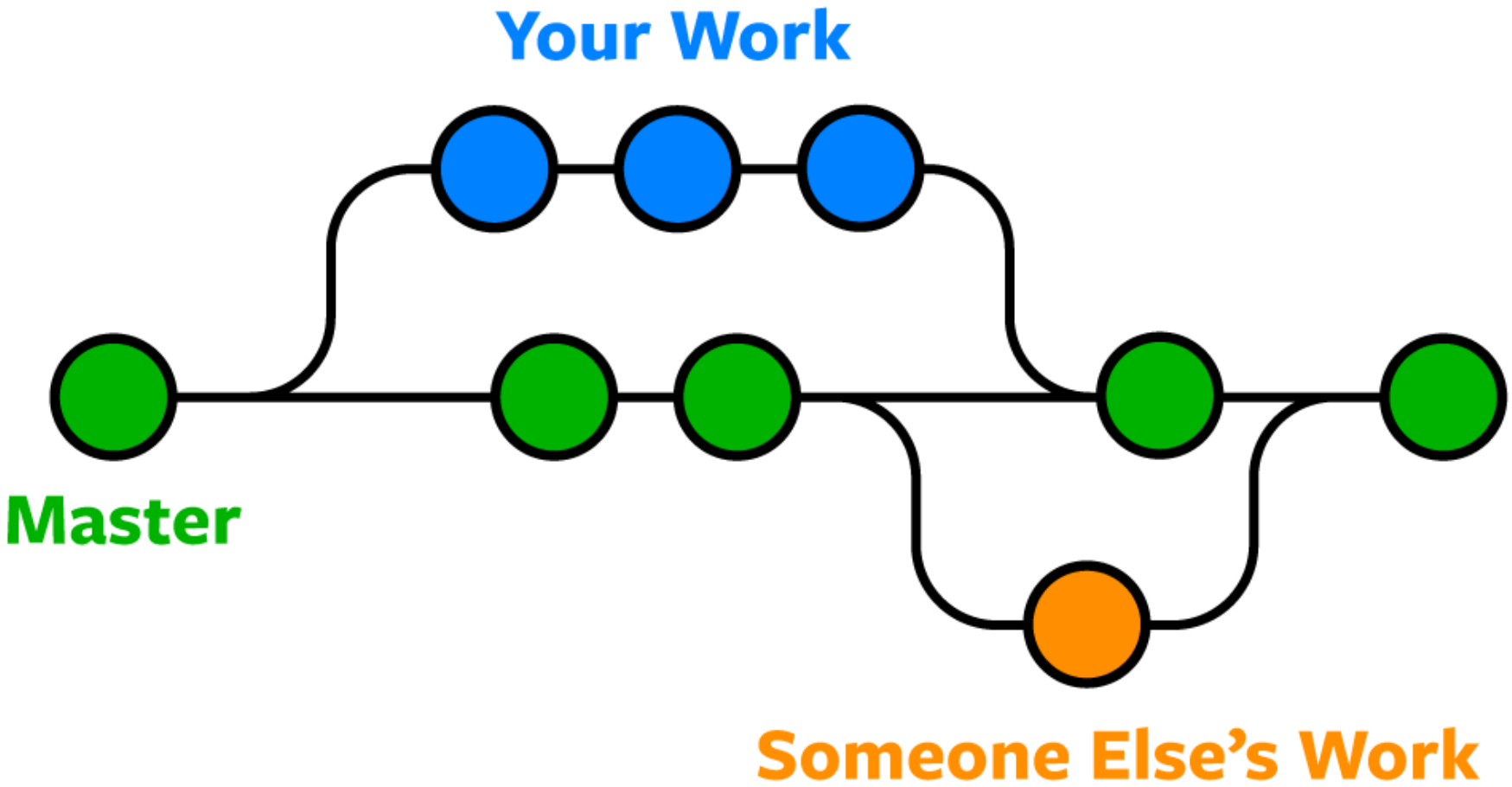


# Key concepts

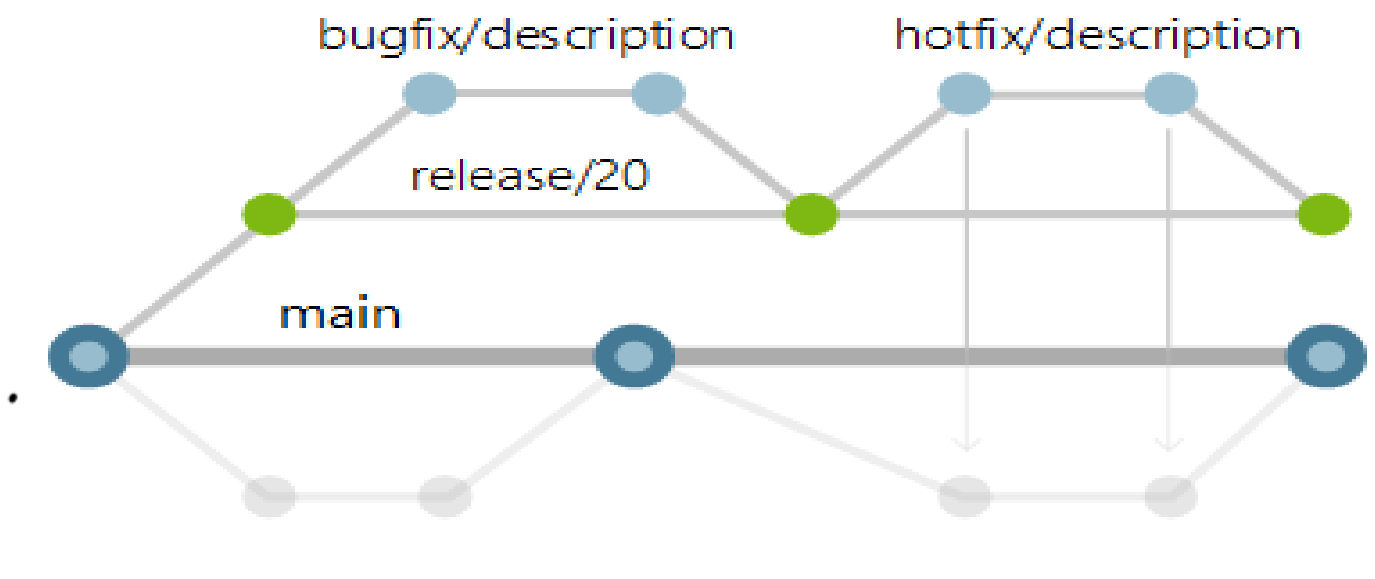
- Repository
- Remote
- Clone
- Working directory
- Branch / checkout
- Commit
- Push
- Pull



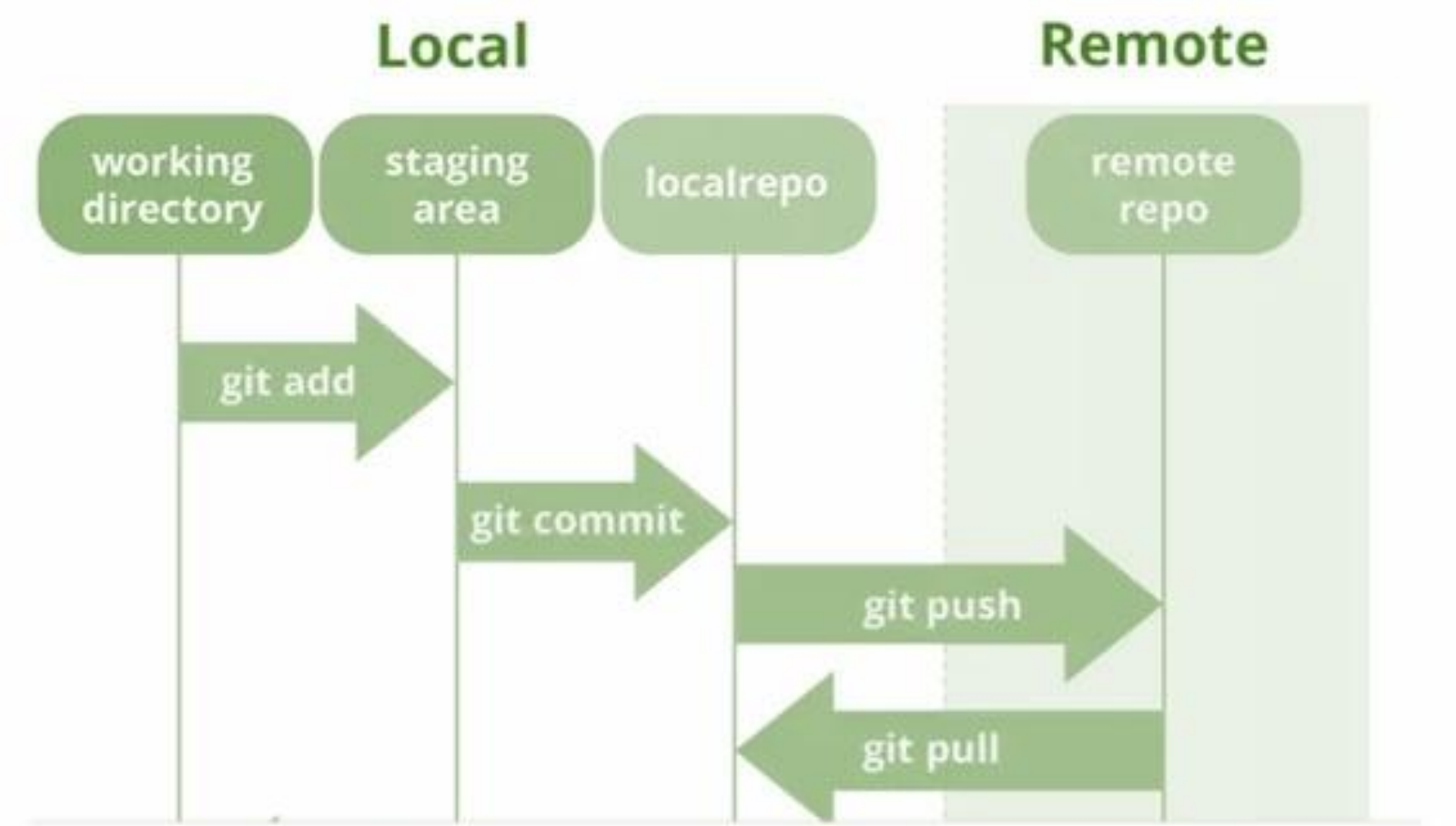
# Branching, commits and merging



# 'checkout' a branch, work on it, add you files, commit your work, push to the remote



# Push and pull to the remote repo to share work with other developers




https://www.janbasktraining.com/blog/git-push-vs-git-push-origin/

# Summary – Lab 2
# In pairs or groups of 3

- ONE person creates a single repository in github to share with the group
- EVERYONE clones the repository
- EVERYONE checks out the 'main' branch
- EVERYONE adds a new file to main, commits and pushes.
- EVERYONE does 'git pull' to get eachothers work.

```
git config --list
```

# Step 1: Create repository and clone it

- You can create your local repository in one of two ways:

- 1. Initialise an empty repository using `git init` in a folder and then push this to your remote.
- 2. Create a repository on the remote platform (Github), then `git clone <url>` to clone this to your local machine (this is what you will do today)

# Step 2: Choose or create a branch to work on

When you clone a repository you will get access to ALL of the branches.

By default, the initial branch is called 'main'.

- git checkout main  (will ensure you are on main)
- git checkout –b mybranch (will create and switch to a new branch)

# Step 3: Modify and create files, and stage them

Now you have a branch checked out into your 'working directory'.

You can modify and add files as you like.

When you have finished some work and you would like to ensure the version you have right now is in the repository you can check what files have changed or been added with the command:

- git status

When you have identified the files you want to keep, add them with

- git add <file or directory name>

# Step 4: commit and push

Your files are now 'staged' for a commit.  You can check them using

- git status

If the correct files are listed, its time to commit them

- git commit –m <message>

To transfer them to the remote repository for others to use do:

- git push

# NEXT WEEK

**Lab**

- Creating a consistent development environment with Docker

**Seminar**

- Group forming and initial discussions
- Group dynamics and code of conduct