Lecture 1

Course info and rules, intro to web apps and REST APIs

Team

- Lect. Dr. Ionescu Vlad-Sebastian - course leader, vlad.ionescu@ubbcluj.ro
- Lect. Dr. Mircea Ioan-Gabriel, Morărescu Cristina, Cărăușu Catrinel, Dragomir Andra - Lab instructors
- You can contact us on MS Teams - join the team using code **dhe2xoo**
- MS Teams is where any announcements will be posted - make sure you pay attention to it

Rules - participation and attendance

- You must attend labs with your assigned formation exclusively. No changes or transfers are accepted except for participation at faculty-sanctioned events. Contact me if you think you qualify.
  - Students retaking the class and students taking the class as an elective can choose the formation they attend with, but they cannot change it after week 2.
- Attendance requirements are as they appear in the faculty rules.
- I will record the lectures, but I will only post the videos if enough students attend them.
- You can turn in at most **two** assignments per week.
- The work you turn in must be your own, and you must be able to explain it.
- Make sure you listen to your lab instructor's instructions carefully, as they might change the posted requirements slightly.

Rules - grading

The following points are available at different stages of the semester:

- Weeks 1-4: up to 4 points
- Weeks 5-8: up to 2 points
- Weeks 9-12: up to 2 points
- Weeks 13-14: up to 1 point
- Examination sessions: up to 3 points
  - Up to 1 point from a theoretical exam
  - Up to 2 points from a practical exam

You need at least **5** points to pass. Everything is rounded down: 4.99 points is a failing grade.

You can stop participating at any time and keep your existing points: **int(points)** will be your final grade. Just let us know if you do this. No takebacks!

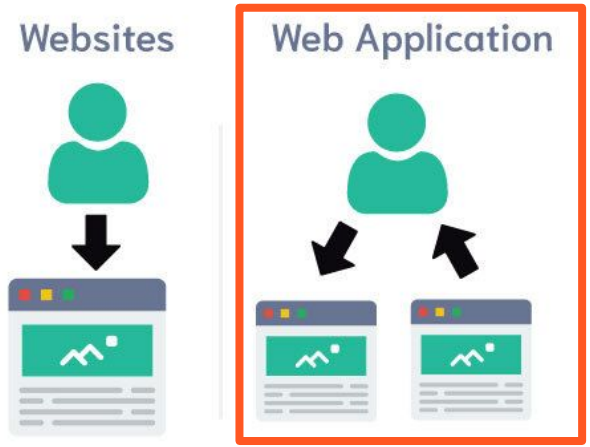
The assignments will get progressively harder.

Principles and philosophy

- This is not a fundamental subject, so it's OK if you're just not that into it
  - It will be easy to get a grade of 5 or 6 by applying your knowledge of basic programming: think FP / OOP level of assignments complexity.
  - It will be harder to get more, and very hard to get a 10. You should only aim for this if you are passionate about the subject.
- You are year 2 students and deserve more independence
  - You can choose your own programming languages and libraries for the assignments - I'll keep some things language agnostic during the lectures and where that's not possible, I'll try to mention alternatives to the language I'll be showing examples in.
  - You'll have less exact examples and more freedom in how you do things. You'll be reading documentations a lot.

What are we doing here?

WEB APPLICATION VS. WEBSITES

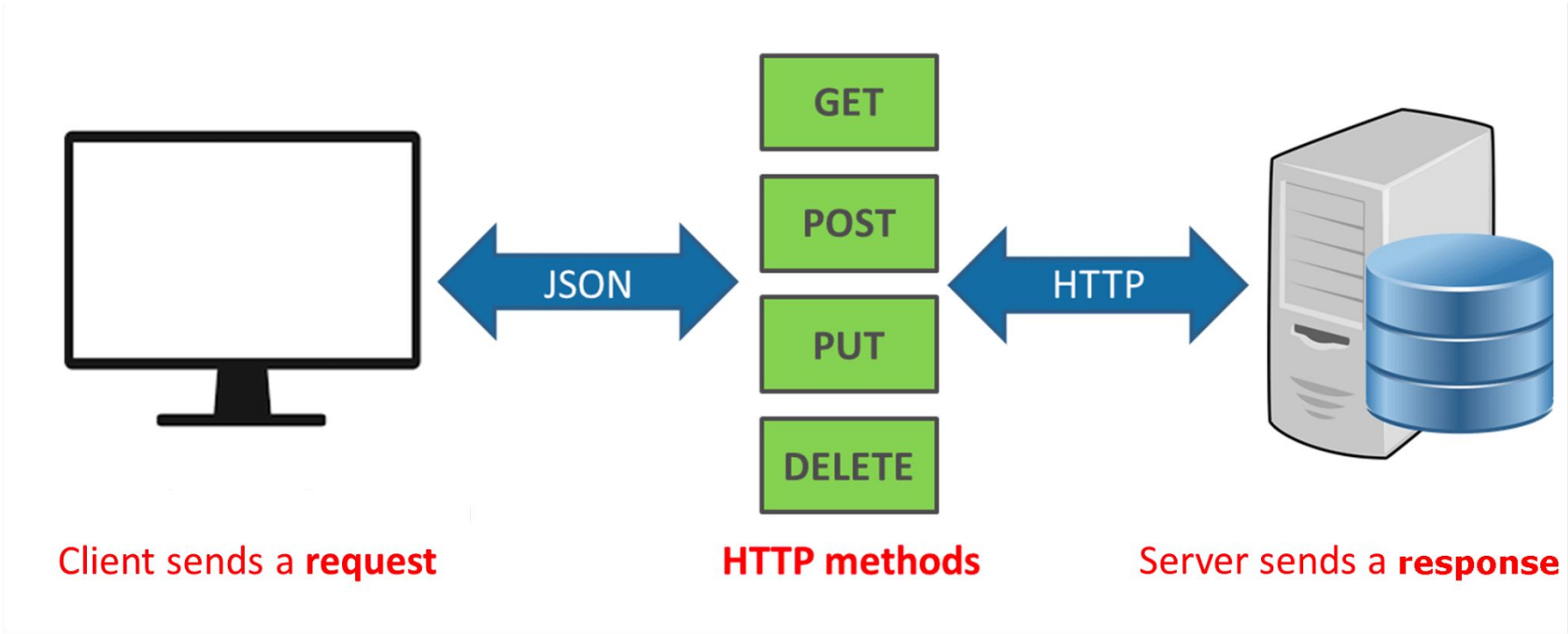


WEBSITE	WEB APPLICATION
A website may or may not require user interaction.	A web application is a web-based application that involves user interaction at every phase
A website is only server based.	A web application is client server based.
A website can be of a single page or multiple pages that are well visible on the home page.	A web application page is navigational, i.e., action required from the next page's user side.
A website can never be a desktop Application.	A web application can be said as a desktop application with a web interface.
A website can be operated without any signup or login requirements.	A web application, most of the time, needs to sign up or log in.

Our take on web applications

- We will create web applications to train as Full Stack Developers.
  - You will learn how to handle most aspects of web app development, mainly:
    - Backend
    - Frontend
    - Database
    - Deployment
    - Security
    - Working in teams
  - We will start with backend frameworks and continue with frontend libraries and frameworks.
- We will create REST APIs that will be consumed by a frontend Javascript-based application.

REST APIs


GET
POST
PUT
DELETE
JSON
HTTP
Client sends a **request**
HTTP methods
Server sends a **response**

REST APIs - read more about it

- I told you you'll be reading a lot of documentation. There's no point in me reproducing what's already all over the internet. I'd rather have a discussion and a debate about the best way of doing things.
- For example, this is a good resource on how it works: https://restfulapi.net/
- Don't worry if it doesn't make much sense at first, it will with a bit of practice.

How do we write code for this?

- We'll be ditching console apps right from the start.
- You'll pick your favorite "Web App" / REST API library from your favorite language.
  - We'll need a REST Framework quite soon, so don't get too used to a "Web App" template
  - I'll pick Django and Django Rest Framework, but I won't be writing much code anyway
  - For Java, you have Spring and Spring Boot
  - For C#, you have ASP.NET Web API
  - For Javascript you have Node
  - For others you have Google
- Install Postman to work with your API.
- Study your framework and complete the first lab assignment.

Examples and terminology

The following should be common across most frameworks:

- MVC / MVT (Model View Controller / Model View Template) - the architectural style most commonly used in web frameworks:
  - **Models** are your domain entities and they store the data passed between the client and the server.
  - **Views** (templates in Django) are the presentation layer and handle user interaction logic; they are absent in REST Frameworks.
  - **Controllers** (views in Django) handle the code that gets execute when a route is requested by the client (www.example.com/students).

Let's look at some frameworks

- Java Spring and Spring Boot
  - https://spring.io/
- Django
  - https://www.djangoproject.com/
  - https://www.django-rest-framework.org/
- ASP.NET MVC and Web API
  - https://dotnet.microsoft.com/en-us/apps/aspnet/mvc
  - https://dotnet.microsoft.com/en-us/apps/aspnet/apis
- Node
  - https://nodejs.org/docs/latest/api/synopsis.html

Worked examples

See any files posted on github:

https://github.com/orgs/UBB-SDI-23/repositories

Get familiar with git, as we'll also be using its more advanced features soon:
https://git-scm.com/docs/gittutorial