IDATA2301 Web Technologies

Course plan, Spring 2022

Version history

v1.0 Initial version, 2021-01-05

v1.1 Some typos and corrections for learning outcomes, 2021-01-07

Learning outcomes

Skills - at the end of the course students will be able to:

1. **Create** a simple **website** according to a given specification
2. Understand and **extend** code of **an existing website**
3. **Design** a webpage **layout** based on a specification
4. Implement interaction using **JavaScript**
5. Implement data **exchange with a backend** (API)
6. Implement **authentication**
7. **Present** their ideas, knowledge, and results

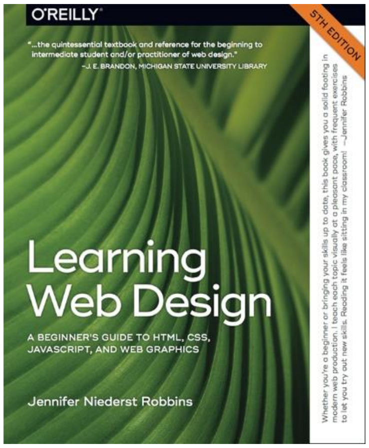
Knowledge - students will understand:

8. The **history** of web application development
9. The web architecture, main functions, and concepts of the different **web solution components**:
   a. Web servers, HTTP protocol and web browsers
   b. Languages: HTML, CSS, JavaScript
10. **Event-driven** nature of websites
11. **JavaScript** language and its use for dynamic websites
12. **Reusable components** for websites
13. The main features of **JavaScript frameworks** in general and **ReactJS** in particular

In addition, students will strengthen the following skills learned in previous courses:

14. Agile work methodology
15. Version control (GIT)

Literature


O'REILLY
5TH EDITION
"...the quintessential textbook and reference for the beginning to intermediate student and/or practitioner of web design."
—J. E. BRANDON, MICHIGAN STATE UNIVERSITY LIBRARY
Learning Web Design
A BEGINNER'S GUIDE TO HTML, CSS, JAVASCRIPT, AND WEB GRAPHICS
Jennifer Niederst Robbins

Jennifer Niederst Robbins, "Learning Web Design", 5th edition.

https://learningwebdesign.com/

Note: the book does not cover Javascript.

Week Plan

In a separate document.

Practical course implementation details

Groups

Students work in groups of 3-4 students. No individual or 2-student groups allowed.

Groups write a contract at the beginning.

Weekly exercises (learning activities)

There will be learning activities (exercises) defined every week. **Students need to do at least 50%** of the weekly exercises to get "that week approved". This is counted individually. **Students need to get at least 60% of all weeks "approved" to get access to the exam (9 out of 15 weeks)**.

There will be multiple pieces of exercise each week, students can choose what they prefer to do.

When physical teaching will be possible, many of the exercises will be done in class, during the live-coding sessions.

Attending lectures and/or watching course videos is considered as an exercise.

The deadline for doing the exercises is Friday 16:00. No extensions. The idea – there is no point of postponing activities, it gets only worse. It is important to learn during the semester, not two weeks before the exam. A "small push" regularly is important, especially during the pandemic.

The check of the exercises will be based on trust – auto-graded test on Blackboard every week where the student specifies what he/she has done this week.

Experienced students

Students who have web-development experience are expected to know the same concepts as all other students, and hence will get the same questions in the oral exam. However, to have some fun during the semester (instead of repeating what you already know and not learning anything new), experienced students can have a custom agreement in the course:

- The student can choose any topic of interest they want to learn instead for basic web development. For example, research a new web framework.
- Instead of doing the proposed exercises, students can set up their own schedule.
- Progression must be kept each week. Each week the student hand in a short description/reflection of what they have learned this week. Teacher must approve whether that seems appropriate level of commitment.

Possible extra topics

There are many extra things for web which don't have enough space in the course. These could be explored by students who have good background in Web development and need extra challenge. The topics include:

- SCSS and Sass for reusable and more structured stylesheets
- Server-side rendering
- PHP or other solutions for a monolith web site solution

- Server-side template languages (Twig, etc)
- NodeJS and server-side javascript
- Bootstrap CSS framework
- Next.JS, Remix.JS, Svelte or other JS frameworks

Checkpoints

There will be multiple checkpoints during the semester. Each group meets with the teacher and have a 15-20min discussion. During this discussion:

2. The group needs to convince the teacher that
    a. All team members are “up to speed” and have control over the topics discussed so far in the course
    b. There is a progression towards the course project
3. If it is discovered that the team needs help, a meeting is set up
4. If a team member is not present during the meeting without a good reason:
    a. First time – a warning is given
    b. If it happens again - the member is discarded from the team and hence cannot deliver the course project (can still participate in the individual oral exam)

Project

Performed in groups.

Several versions possible, with different levels of complexity (and grade):

1. **D-level project**: a presentation website for a local company (small shop). Mostly static, loads news articles from a backend. No authentication.
2. **A/B-level project**: a dynamic website for a local company with a simplified online shop (users can see and comment products, can add products to a cart and send in an order, no payment) with authentication and two user groups:
    a. Customer, can do the following:
        - Signup (email-confirmation is optional)
        - Login
        - Select products
        - Add products to shopping cart
        - Send in an order
        - Write a comment for a product
        - Edit her own comments
        - See other user comments
    b. Site owner (admin), can do the following:
        - Add new products (pictures can be added as URLs to external images)
        - Update products
        - Delete products
        - See all user orders
        - Mark an order as processed
        - See all user comments for each product

        - Delete any user comment
3. **Custom A/B-level project**: develop another rich web-application for a domain of your choice. The application must be at least as complex as the other A-level project. Custom projects must be approved by the teacher.

For the first two options teachers provide a backend-solution that can be used. For the third option students must develop a backend themselves.

Ideally frontend part is delivered as project in this course while backend part is delivered as the course project in IDATA2306 Application development.

Non-functional requirements for any project:

- Responsive design.
- Follow universal design (accessibility) principles.
- Look aesthetically pleasing, according to current “website trends” (not necessarily very creative, but not *ugly*).
- Each group gets a different genre of the shop (teachers provide a selection of branches, such as eco products, shoes, toys for toddlers, bicycle rent, hiking equipment, *idrettslag,* sports outlet, men’s suits, cosmetics, etc). The overall style of the website should match the style of the shop (serious VS funny, etc).
- All code must be stored in Git, throughout the whole semester. Git log is part of the final delivery.

Project delivery

The following deliverables are handed in as the "course project" in the end:

1. A short video presenting the project (10-15min). The video must show the product and convince that students have worked professionally (i.e., reflection on both the used concepts and the development process)
2. Source code of the project
3. A link to a publicly accessible website in operation (the site must be deployed on a publicly available server)

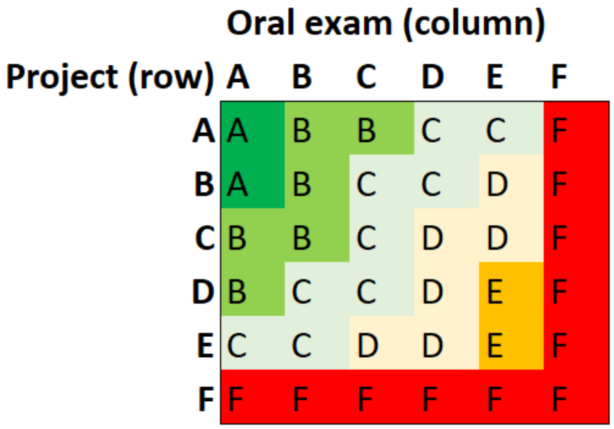
Exam – grading

The final grade in the course is based on two parts:

- 40% Project in groups, see project delivery
- 60% Individual oral exam, see document about the oral exam

The project evaluates student ability to create a real website in a team. The oral exam checks individual level of understanding.

The final grade is calculated, using the following matrix:



	Oral exam (column)					
Project (row)	A	B	C	D	E	F
A	A	B	B	C	C	F
B	A	B	C	C	D	F
C	B	B	C	D	D	F
D	B	C	C	D	E	F
E	C	C	D	D	E	F
F	F	F	F	F	F	F