

# IMY 210

## Project: Scheduling application

### Important!

- Make frequent backups of your project files, preferably in different locations
  - **Recommendation:** create a repository and use some form of change tracking software (GIT)
- Ensure you are using version 1.0, no encoding is required.
  - You may use any instructions/functions you wish, even those we did not cover in class.
  - As long as those functions fall under version 1.0 of the XML languages you are using.
- This is an **individual** project.
  - See the study guide for more info about plagiarism.

### The Scenario

- You are creating a scheduling application for users to keep track of their weekly activities by creating events (**think of any calendar application**).
- Using this as a prototype and not wanting to deal with databases, you decided to **use XML to store the data**.
- To keep the **XML data at a centralise location you choose to use a RESTful server for accessibility**.
- Lastly, users should be able to **push or pull calendars directly from the server**. ✓

### About the system

- You will be creating a **Vue based web application that used XML as the main storage method**. (Frontend)
- The **XML files will then be stored on a RESTful server** that will be directly retrieved by the Vue application. (Backend)
- **Users can specify a specific calendar they wish to access** (be it by name of the user, or id, or selecting from a list)
- **If the calendar belongs to the user accessing the calendar, they will be able to perform basic CRUD operations on events on the calendar, if not, the user will only be allowed to look at the requested calendar**.

### Provided files

- Sample.xml – a simple sample found to keep track of a user's scheduling.
  - You are welcome to add **additional XML** files to your final submission, your XSD validation does not have to validate these additional files.
  - Do not change the format of the sample file that is provided for you.

### Warning!

- This project will require you to learn several methods of integration between systems and languages. Plan and work on this consistently throughout the next two months.
- Ensure that your **details are all located in all relevant files**.
- If your **validation/style sheet documents are not well-formed**, you will be penalised 50% for those sections
- 50% of the section mark constitutes the working condition of a section, the reminding 50% is made up of the technical implementation of the language/technique used to show your mastery of the language.

## Phase 1 – Treating the XML files (35%)

### A) Validation (15%)

Create a **single XML Schema** document called **validate.xsd**.

- This **single** schema document must be able to **validate provided files**.
- Study the XML document provided as the rules will be provided within the document.

### B) Displaying information (10%)

Even as individual XML files, each schedule should be styled and accessible manually.

For simplicity, create a **single XSLT stylesheet to display** the information from the provided XML files **simplistically and legibly**. Ensure that the styling makes sense in the context of your application.

Ensure that this stylesheet is data agnostic and can be used with another subsequent XML files. Remember to **link** all existing and newly created XML documents to this XSLT stylesheet.

**Note:** The styling you provide here should reflect the final styling done in your frontend, use this section to style your XML according and transfer the styling to your final frontend. You can take **inspiration from some existing calendar applications**.

### C) Calendar publishing (10%)

Using your understanding of **XSL-FO** and **EPUB** render a small calendar (pdf) that hold all event for the user. The EPUB should display all the data fields from each event and sort them accordingly.

For the scope of the project, you do not need to auto-generate the final PDF. You can assume that during the marking process, the marker will generate the PDF using the command line/terminal.

## Phase 2 – Preparing the backend (25%)

### D) RESTful server (25%)

Create a restful server that will be responsible for all requests from the frontend.

Your restful server should support the four fundamental calls of CRUD (**create, read, update, delete**). This server should allow you to directly pull the XML data from the server without having to directly access the XML documents. Send the data back to your front end in a manageable manner, you can transfer this data in any manner you wish but remember your final storage point is still XML.

**Note:** Keep in mind that you will be running two servers, one is your RESTful server (database) and one is your frontend Vue application. Make sure you do not mix up the port each of these services is running on. Remember that you can use Postman to test your RESTful server directly.

**Important:** If you are unable to set up your server, you are welcome to set up a simple state management file for your frontend and work using that shared data file. You will however lose the marks for this section. [prac3](#)

## Phase 3 – Completing the frontend (40%) [vue](#)

### E) Setting up the application (10%)

Take your styling done in section two of the project and **fragment your final design into several components**. **Create your components** and ensure that all **additional libraries** needed by your application are **referenced correctly**.

You should also take this opportunity to **ensure that all the extra functionality (create, update and delete) has been included in your design**.

### F) Interacting with the data using functions (20%)

Ensure that your application is communicating with the server correctly and **using the data from the server to populate your final application**. [only if you have a RESTful API ????????](#)

You will need to **link functionality where needed: Creating, Retrieving, Updating and Deleting**. [prac5](#)

### G) Look, feel, integration (10%)

A portion of the mark will be given on how well you design your system. If you are unsure about the principles of design, please look at “Dieter Rams: 10 Commandments for Good Design” as a guideline. Your final design should be **user friendly, easy to use and self-explaining**.

Ensure that the **final data you have sent with your final submission includes:**

**5 users' schedules**

**Each user should have at least 10 events on their respective calendar.**

### Bonus

The following section consists of tasks you can attempt for additional credit. Each task can vary in weighting from 2% to 5% depending on the difficulty of the implementation.

- Auto validation
  - When an XML file/format is pushed to your server, the script within your RESTful server automatically validates your XML file using your XSD document.
- File export
  - Allow the user to have the function of exporting the XML and XSLT transformation to be a regular HTML page
- RSS Notification
  - Whenever a new event is created, an RSS feed will be updated. The RSS feed will hold five of the latest events that have been added to the system. Users can register this RSS feed using any existing aggregator (you can simulate this locally).

### Submission

- Double-check that your full name and student number have been added to all your files, and your XML-based files are well-formed.
- Include two folders in your final submission
  - Folder “Vue” will contain all your code for your frontend
  - Folder “RESTful” will contain all your code for your backend
- **When uploading your final project remember to exclude both node\_modules folders.**
  - **Warning:** You will be penalised if you include this in your final submission.
  - **Note:** Move your files to a new location and run the *npm install* command to reinstall the modules needed and test your application.
- Include a short **readme file** about what has been done in the application and what users should do to get your application to run.
- Compress both these folders into a ZIP archive name **Project\_studentNumber.zip**
- Make a final backup of all your files and keep it in a safe place.
- Submit your **ZIP** file to the upload slot provided on clickUP before the deadline.
- **Note:** If you submit your final submission to the **late** deadline slot, you will have an additional few days to complete your project at the cost of **30%** from your final mark.