FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Frontend Application Tasks – Web Engineering

General Requirements (5 Points):

> Responsive page content (at least: 1920×1080, 1440×900, 360x640)

> **No Javascript Frameworks/Libraries** (JQuery, Angular, React, Vue and so on) – CSS Libraries are ok -> **K.O. criterion!**

> Clear and consistent project structure (e.g. folders, resources, …)

> Separate resource files (e.g. css, html and scripts)

> WAI compliant (Lighthouse analysis)

Learning Objective: Know how to setup a web project using proper build and dependency management.

Project setup and build management (10 Points)

> Build the application with npm and a build and dependency management tool of your choice (e.g. Vite, Webpack, or others)
> Define 2 tasks in npm scripts:
    > Development
    > Production build (e.g. obfuscated, minified, bundled)
> Keep your builds clear and add dependencies to the right build (e.g. do not add dev dependencies inside the production build)
> Configure your project to use Typescript as your primary development language.

Learning Objective: Create a basic HTML5 project with navigation

Multiple Pages (e.g. Home, About, Contents, …) (5 Points)

> Implement multiple pages with proper navigation

> You can use a single-page (dynamic content reloading) or a multipage approach

Learning Objective: Know how to integrate APIs via Ajax from JS/TS

Consume API Data using Ajax (10 Points)

> API samples:

- List public APIs: https://github.com/public-apis/public-apis/blob/master/README.md
- https://english.api.rakuten.net/matchilling/api/chuck-norris
- https://http.cat/
- https://dog.ceo/dog-api/documentation/

You can decide HOW your app consumes the API data (XMLHttpRequest, fetch, other libraries)

Learning Objective: Use the DOM API to manipulate the UI dynamically

DOM Manipulation (10 Points)

> Render requested API data in HTML (table, list, containers, …)

> Users can **add and remove** items from previously loaded data to another element (e.g. "favorites" or "shopping cart")

Learning Objective: Know how to implement HTML5 forms an how to properly validate user input with JS/TS and HTML5

Implement a form (10 Points)

> that consists of following fields: **text, number, password, email**

- make use of proper html attributes!

> Input is validated on user event (click, hover, enter, …). Choose at least 1 event handler. Input should be validated via **JS/TS and HTML5**.

> Input validation error/success is presented to the user (modal, in-form highlighting, …)

Learning Objective: Know how to implement a CI/CD pipeline to automate the development and deployment process – write automated tests

CI/CD (20 Points)

> Write at least **2 unit tests** for your project and configure tasks in npm scripts (5 points)

> Configure a **linting** tool of your choice and add it to your npm scripts (5 points)

Configure 2 Workflows in GitHub Actions, one for development and one for deployment (10 points):

> Create a `development` Branch inside your repository

> **Development Workflow** should at least test and lint your code when developers push to branch `development`

> **Deployment Workflow** is triggered when developers push into `main` branch. It should at least test, lint and build your source code. Afterwards the build artifacts of your application should be automatically deployed to Github Pages (or another hosting provider of your choice). Note: in Github Actions it is possible to reuse existing workflows in other workflows.

Overall Points: 70