Web Application Development (LTAT.05.004)

COURSE SUMMARY, EXAM PREPARATION, STRUCTURE & METHOD

MOHAMAD GHARIB
UNIVERSITY OF TARTU
Course summary
Course summary
Course summary
Course summary
Course summary
Course summary
Course summary
Course summary
Course summary - Homework
Course summary - Homework
Course summary - Homework
Course summary - Homework
Course summary - Homework
Course structure

- WAD basic concepts
- JavaScript
- Vue.js + Vuex
- Node.js (DB + security)
- Testing
- HTML
- DOM + AJAX
- Homework #3
  Vue + Vuex
- Homework #4
  Node.js + DB + security
- CSS
- Homework #2
  Javascript
- Homework #1
  HTML + CSS
- EXAM
Exam structure
Exam structure
Exam structure

- Coding exercise(s) (HTML and CSS) 10 points
- Back-end
- SQL
  - INSERT
  - SELECT
  - UPDATE
  - DELETE
- Database
- Cypress
- Vue
- Svelte
- Node.js
- PostgreSQL
Exam structure
Exam structure

Coding exercise(s) (Cypress) 10 points

SQL
- INSERT
- SELECT
- UPDATE
- DELETE

Coding exercise(s) (Vue.js, Node.js, db. jwt) 30 points

Back-end

Coding exercise(s) (HTML and CSS) 10 points

end

Database
Exam method
Exam repo import

Project Setup:
You need to import this <repository> to your GitHub account

Note: you have to be logged in to your GitHub account to do the following steps

• In the first field "Your old repository’s clone URL" enter the URL of this repository https://github.com/M-Gharib/WAD2022-Exam-Example.git - just an example of the exam
• In the "Your new repository details", give the repo a name (e.g., WAD22-Yourname), and make it Private.
• Press on "begin import", it should take a few seconds
• Add me as a collaborator to your new repository (inviting collaborators). My email and username on GitHub are: mohamad.gharib@ut.ee and M-Gharib respectively.
Exam repo import

Project Setup:

You need to import this repository to your GitHub account

Note: you have to be logged in to your GitHub account to do the following steps

- In the first field "Your old repository’s clone URL" enter URL of this repository  https://github.com/M-Gharib/WAD2022-Exam-Example.git
- In the "Your new repository details", give the repo a name (e.g., WAD22-Yourname), and make it Private
- Press on "begin import", it should take a few seconds

Note: Please do not invite me as a collaborator for this example exam, just do it for the real exam.

- Add me as a collaborator to your new repository (inviting collaborators)
  - My email and username on GitHub are: mohamad.gharib@ut.ee and M-Gharib respectively
Importing a repository with GitHub Importer

Information on how you can import a repository using the GitHub Importer can be found [here](https://docs.github.com/en/get-started/importing-your-projects-to-github/importing-source-code-to-github/importing-a-repository-with-github-importer)
Inviting collaborators to a personal repository

Information on how you can invite collaborators to a personal repository can be found [here](https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository)
Exam App
Important notes

• Commit and push changes to your repo after finishing each task.
• Do not forget to commit and push your changes, when you are done.
• Do not make any changes to your repo after the end of the exam.
• You may not be able to copy and paste into the text area for answering this exam, but you can add attachments (up to two). Therefore, add your name, code, and a link to your exam repo within this attachment (.txt).

Note: if we observe suspicious behaviour, the student involved will be called for an interview. Examples of suspicious behaviour can be, all the commits happening at the end of the exam or with a non-realistic timeline, too many similarities between projects, etc.
Important notes:

- Commit and push changes to your repo after finishing each task.
- Do not forget to commit and push your changes, when you are done.
- Do not make any changes to your repo after the end of the exam.

You may not be able to copy and paste into the text area for answering this exam, but you can add attachments (up to two). Therefore, add your name, code, and a link to your exam repo within this attachment (.txt)

Note: if we observe suspicious behavior, the student involved will be called for an interview. Examples of suspicious behavior can be, all the commits happening at the end of the exam or with a non-realistic timeline, too many similarities between projects, etc.
App description:

1. Table 1 shows table "routes" that is created automatically in your database, also 6 records have been inserted into it. In database.js that is in the ../server directory, you can find information related to the table and inserted records.

<table>
<thead>
<tr>
<th>TABLE &quot;routes&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>fromcity</td>
</tr>
<tr>
<td>tocity</td>
</tr>
<tr>
<td>cost</td>
</tr>
<tr>
<td>departuretime</td>
</tr>
<tr>
<td>departuredate</td>
</tr>
</tbody>
</table>

2. The backend app (Node.js) can deal with three different requests (GET, POST and DELETE). In server.js that is in the ../server directory, you can find more information related to these requests.
Tasks – Frontend

Tasks:

**Task 1.** Modify "AllRoutes.vue" to fetch the content of the "routes" table through a GET request and present it as shown in Figure 1. [X point]

**Note:** the back-end can deal with this Get request, your code should be only in AllRoutes.vue.
Tasks – Frontend

**Task 2.** Add the required styling (CSS) to make the page looks like Figure 2. [X point]

![Figure 2.](image)

**Routes | Routes Management**

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Cost</th>
<th>Time</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tartu</td>
<td>Tallinn</td>
<td>16</td>
<td>06:00:00</td>
<td>2022-03-24</td>
</tr>
<tr>
<td>Tartu</td>
<td>Tallinn</td>
<td>14</td>
<td>08:00:00</td>
<td>2022-03-24</td>
</tr>
<tr>
<td>Tartu</td>
<td>Parnu</td>
<td>11</td>
<td>10:00:00</td>
<td>2022-03-24</td>
</tr>
<tr>
<td>Tartu</td>
<td>Narva</td>
<td>15</td>
<td>10:30:00</td>
<td>2022-03-24</td>
</tr>
<tr>
<td>Tartu</td>
<td>Tallinn</td>
<td>12</td>
<td>11:00:00</td>
<td>2022-03-24</td>
</tr>
<tr>
<td>Tartu</td>
<td>Parnu</td>
<td>12</td>
<td>12:00:00</td>
<td>2022-03-24</td>
</tr>
</tbody>
</table>
Task 3. Modify your code so that when the cost of a trip is more than 12, its background will be red. Otherwise, its background should be blue as shown in Figure 3. [X point]
Tasks – Frontend

Task 4. Add an element that represents the number of trips as shown in Figure 4. \[X point\]

Note: the shown number should be dynamic (not static), i.e., it should count the number of offered trips and update the value when trips are added or removed.
Tasks – Frontend – Another example

You might be presented with how the frontend looks and asked to code the functionalities and styling.

Task 1. RouteManagement.vue (shown in the Figure below) is supposed to fetch and represent all available routes in the database. [X point]

Task 2. Each route should be represented as a row in a table accompanied by two buttons (update and delete). [X point]
Tasks – Backend

Backend examples:

Task 1. Modify the put request in server.js in a way that allows modifying only the time and date of a trip. [X point]

Task 2. Write a Get request handler in server.js that takes a city name as a route parameter and returns all trips departing from that city. [X point]

Task 3. Write a Delete request handler that deletes all records in the database. [X point]

Task 4. ....
Tasks – Backend examples:

**Task 1.** Modify the put request in server.js.

**Task 2.** Write a Get request handler in server.js departing from that city. [X point]

**Task 3.** Write a Delete request handler that...

**Task 4.** ....

Another Backend example: (you might be presented with the front end App and asked to code the functionalities of the backend)

RouteManagement.vue (shown in the Figure 6) is supposed to fetch and represent all available routes in the database. It can also modify or delete any route, and add new routes through the form.

**Task 1.** Write a Get request handler in server.js that fetches all available trips from the database [X point]

**Task 2.** Write an update request handler in server.js that updates a specific trip as shown in the figure [X point]

**Note:** the update button will trigger the update the updateRoute function, which you need to complete to provide the required functionality.

**Task 3.**....

Figure 6.
Tasks - Testing

Testing: (Either you will be asked to write a test for a component/API, or you will be provided with a test and asked to make it pass by modifying your code)
Test cases can be found in cypress/e2e/exam1-tests.cy.js

Test 1. Do the required modification to "RouteManagement.vue" so that test 1 passes. [X point]

Test 2. Write the required code in Test 2 so that it tests adding a new route through RouteManagement.vue. [X point]

Test 3. Write the required code in Test 3, so that it tests a route post request to your endpoint. [X point]
Tasks - Testing

Testing: (Either you will be asked to write a test for a component/API, or you will be provided with some test cases to modify your code)

Test cases can be found in cypress/e2e/exam1-tests.cy.js

Test 1. Do the required modification to "RouteManagement.vue" so that, test 1 passes. [X pending]

Test 2. Write the required code in Test 2 so that it tests adding a new route through RouteM...
Test the *shared exam example*, and remember you need to have *Postgres* installed on your Laptop.

*Time* is part of the *exam*, *use it wisely*
Good luck

Questions?
Thank You
for your attention

Mohamad Gharib
mohamad.gharib@ut.ee