|  |  |
| --- | --- |
| Kristianstad University  SE-291 88 Kristianstad  +46 44-250 30 00  www.hkr.se |  |

Bachelor of Computer Science in Education  
Semester Year. Spring Semester 2022  
Faculty of Natural Sciences

Lab 1

NodeJS

Kiasar “Kia” Mian

Content

[Database 3](#_Toc100261758)

[Tasks 3](#_Toc100261759)

[All Users 3](#_Toc100261760)

[Specific User 3](#_Toc100261761)

[Create User 3](#_Toc100261762)

[Update User 3](#_Toc100261763)

[Delete User 3](#_Toc100261764)

[API 3](#_Toc100261765)

[Heading level 2 (style Heading 2) 4](#_Toc100261766)

[Heading level 3 (style Heading 3) 4](#_Toc100261767)

[Fixing the table of contents 4](#_Toc100261768)

[What do you mean headings 1, 2 and 3? 4](#_Toc100261769)

[Page numbering 4](#_Toc100261770)

[If you have problems using the template 5](#_Toc100261771)

[Make sure you are using the latest version of Word 5](#_Toc100261772)

[If you prefer another software for writing 5](#_Toc100261773)

[If you need help 5](#_Toc100261774)

# Database

For this lab I decided on MongoDB. The document structure that was decided on is as follows:  
  
\_id String  
name String  
age Int32

# Tasks

For this lab I decided to use *dotenv* as well as *ejs* to allow for easier and better handled dynamic loading of page routing and content.

## All Users

Fetching from */api/users/* each account object is then looped through and the div *user\_list* gets populated with content. It also assigns the user row an ID that matches the users id.

## User Details

This function makes use of the *this*  property. Since the button has a data-id matching the users id this is used to fetch the correct user from the database using a jQuery built in method *.attr*. The fields are then targeted and the name and age is rendered for that specific user.

## Create User

This function targets the two input fields dedicated for creating a user. The values *name & age* are collected and then passed on to the backend that handles the generation of an ID.

## Update User

The update user section is in two parts. First the user pressed the *pencil* icon of the user they wish to edit. That users information is the populated into the user update section. The ID section is auto-filled but also disabled for a reason, this is to ensure that a incorrect ID isn’t passed.

## Delete User

The delete user like the update and view details function gets the *data-id* through the *this* object, and then sends a fetch request to the backend with the correct ID to delete.

# API

### Create user – POST - /api/users

Text

Description automatically generated

### Delete user – DELETE - /api/users/:id

A screenshot of a computer

Description automatically generated with medium confidence

### Update user – PUT - /api/users/:id

Text

Description automatically generated

### View All – GET - /api/users

Text

Description automatically generated

### View info – GET - /api/users/:id

Text

Description automatically generated