

## Authors:

Richard Langtinn ([r.langtinn@gmail.com](mailto:r.langtinn@gmail.com)) student-id: 512961

Ola Hulleberg ([ola.hulleberg@gmail.com](mailto:ola.hulleberg@gmail.com)) student-id: 498711

**Ola - Did everything (client communication, server communication, latency calculation code, API, and database+Heroku configuration) except style the forms**

**Richard - Heroku setup, forms, checking latency, testing the application and file-serving architect (Express static pages)**

## Documentation:

Our processing layer consists of an endpoint running express with static pages served from the client folder (using `express.static()` function), and an API route following the REST-spec: create = POST, read = GET, update = PUT, delete = DELETE.

### **/user** endpoint:

*filter* can contain **any/every parameter** derived from **userModel**, but we only use *student\_id* in our application.

API will return either *HTTP* status **200** (OK) or **500** (BAD) depending on the result.

### **RETURN (every crud operation's return):**

```
{
  user: { userModel[] },
  timestamps: {
    server_received_timestamp,
    server_sent_timestamp,
    server_latency
  },
  error: {}
}
```

### **CREATE - POST with parameters derived from userModel (Schema):**

```
{
  firstname,
  surname,
  student_id,
  age,
  nationality,
  degree
}
```

### **READ - GET with parameter:**

student\_id

**UPDATE - PUT with filter + parameters derived from userModel (Schema):**

```
{
  filter: { userModel[] },
  firstname,
  surname,
  student_id,
  age,
  nationality,
  degree
}
```

**DELETE - DELETE with filter:**

```
{
  filter: { userModel[] }
}
```

**Github:**

<https://github.com/rlangtinn95/cloud-oblig1.git>

**Heroku:**

<https://oblig1ri.herokuapp.com/>

**Latency:**

**Desktop and wi-fi:**

Round-trip Latency: 49ms  
Server Latency: 45ms  
Client Latency: 94ms  
Action: READ

Round-trip Latency: 180ms  
Server Latency: 56ms  
Client Latency: 236ms  
Action: CREATE/UPDATE

Round-trip Latency: 161ms  
Server Latency: 51ms  
Client Latency: 212ms  
Action: DELETE

**Smartphone and mobile connection (4G/5G):**

19:22

obligtri.herokuapp.com

Retrieve Delete

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 533ms  
Server Latency: 48ms  
Client Latency: 581ms  
Action: DELETE

19:22

obligtri.herokuapp.com

Retrieve Delete

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 656ms  
Server Latency: 44ms  
Client Latency: 700ms  
Action: READ

19:22

obligtri.herokuapp.com

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 872ms  
Server Latency: 50ms  
Client Latency: 922ms  
Action: CREATE/UPDATE

## Smartphone and wi-fi:

19:21

obligtri.herokuapp.com

Retrieve Delete

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 93ms  
Server Latency: 49ms  
Client Latency: 142ms  
Action: DELETE

19:21

obligtri.herokuapp.com

Retrieve Delete

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 51ms  
Server Latency: 45ms  
Client Latency: 96ms  
Action: READ

19:20

obligtri.herokuapp.com

Student Info:

First name: Richard

Last name: Langtinn

Age: 21

Nationality: Norwegian

Degree: Bachelor

Create/Update

Round-trip Latency: 215ms  
Server Latency: 50ms  
Client Latency: 265ms  
Action: CREATE/UPDATE

## Conclusion latency:

Server-latency in our situation is to perform actions on the database, it's a database latency. Latency in terms of 4G/5G goes slower because of heroku, but the server-client latency remains the same.