Student: Rupert Gobber
Student ID: 207554

# Introduction to Service Design and Engineering

### Project Report

## Description

The main goal of this project is to create a management of events, in order to make the management of the day and free time easier.

To do this, an event aggregator was developed. In particular, this service asks for events on Google Calendar and aggregates them to Povo free classrooms. In this way it is possible to keep track of events and in which classrooms it is possible to spend free time between events.

The source code of the project is on GitHub at the following links:
- Main project https://github.com/gobberr/ISDEproject
- Easyroom adapter https://github.com/gobberr/easyroom-service

The services are deployed on Heroku at the following links:
- Main project https://isde-project.herokuapp.com
- Easyroom adapter https://unitn-service.herokuapp.com

The whole project is developed in JavaScript, using the framework Node.js. The persistence of data is located on a MongoDB database. All the services are called through APIs. To run the project on your local computer, you need to install dependencies for each service. See the deployment section for more information.

The APIs documentation is available on the Github Wiki for each service. The links are as follows:
- https://github.com/gobberr/ISDEproject/wiki
- https://github.com/gobberr/easyroom-service/wiki

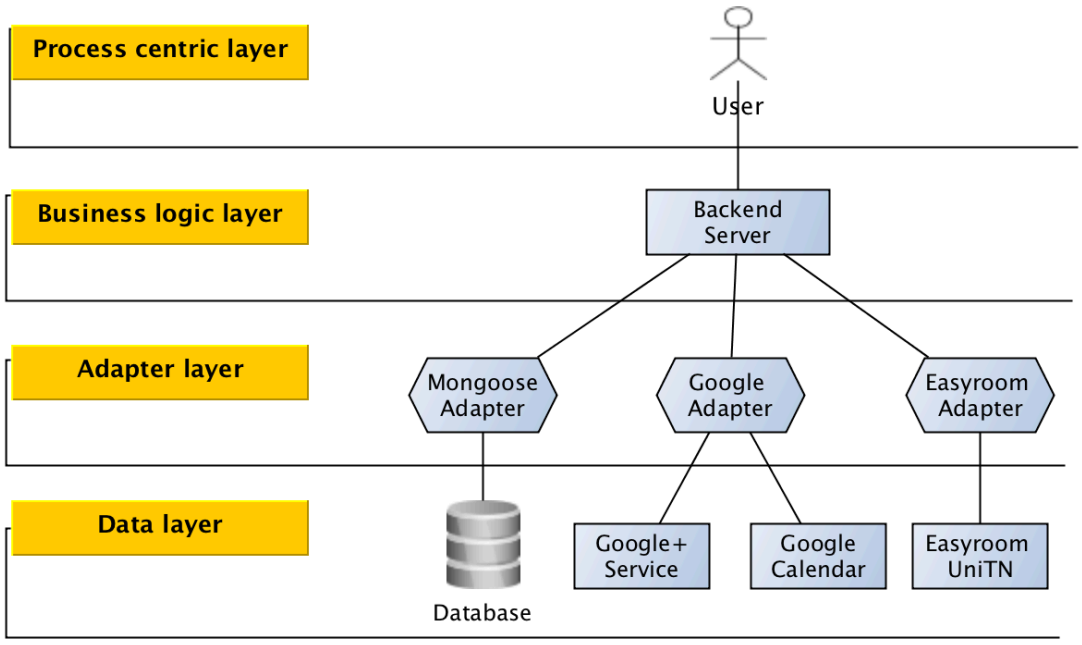
## Architecture

In architecture design, services have been divided according to levels.
In particular, the data layer deals with data management and storage. There is a MongoDB database hosted on mlab.com. The other data come from the Google Calendar, the Google user info and the Easyroom service, which returns the free UniTN classrooms.

3 adapters have been implemented:
- one for database management
- one for the two Google Services
- one for the Easyroom

These adapters manage and filter data that will be processed by the Business Logic level. All user requests are made through a frontend, which routes everything to the Business Logic level.

In the figure we can see the architecture.




## Data layer

- Google+ service, used to log people that use this web service;
- Easyroom service, used to get the occupations of Povo rooms;
- Google Calendar service, used to ask permission to get events in specifics calendars;
- Mongodb service, used to store some data and simplifying the application's workflow.

## Adapter layer

- Mongoose Adapter: this service communicates with a MongoDB storage service through the mongoose.js APIs
- Google Adapter: this service is closely linked to the backend server, as it was chosen that features such as login (with Google+) were integrated in close contact with the orchestration logic

- Easyroom Adapter: this service (https://unitn-service.herokuapp.com) makes a request to the easyroom service and manipulates the data in order to make them compatible and integrable in the merge logic with the calendar events

**Business Logic layer:**

This level uses the JSONs coming from the adapters to process the merge of the events with the free classrooms. Furthermore, it handles requests from the frontend and processes the workflow

**Process Centric layer**

Provides a frontend to the client and routes calls to the backend server. User Interface usage:
There are many pages, described as follow:

- "Homepage", with some informations about the usage of this web applications
- "Login" page, accessible pressing the "Login" button. After Google login, you will rendered in the "Profile" page.
- "Profile" page, where you can see some informations about your account
- "Logout" page, visible after an user is logged, that allows you to close the session
- "Select Calendar" page, where you can insert the name of your calendar. After a calendar is selected, a button "See events" will appear. Press it to see the events retrieved by calendar APIs
  - Note: the blue calendar name is "primary" by default
- "Freerooms" page that call the Easyroom Api and retrieve the rooms occupation of Povo
- "Day Planner" page, that will perform the merge task of events and free room in order to create a table with events and room for each time slot. A time slot duration is half hour.

**Local deployment**

It is necessary to have installed Node.js

- `sudo apt install nodejs`

For each services, follow this procedure:

- `git clone <service-url-github>`
- `npm install`
- `npm start`

See APIs documentation or README.md of each service for more info.