

# main module

**main.add\_module()**

Add module to database. required in POST request body: SmObjId, PiqYear, PiqSession, whitelisted, searchterm

**main.add\_searchterm()**

Add searchterm to DB, term is supplied in form data

**main.admin()**

Administrator front end view

**main.app** = <Flask 'main'>

**main.check\_which\_saved(modules: list)**

Check which modules are saved, and mark them as either white- or blacklisted accordingly

**main.cors** = <flask\_cors.extension.CORS object>

**main.db\_config** = {'database': 'greenvvzdb', 'host': '127.0.0.1', 'password': 'greenvvzpw', 'user': 'greenvvz'}

**main.find\_modules\_for\_course(course: dict)**

Request detail page for course object, add Module subobjects(dict) as list to given course object

**main.find\_studyprograms\_for\_module(SmObjId: int, PiqYear: int, PiqSession: int) → list**

Request detail page for module object, add Studyprogram subobjects(dict) as list to given module obj

**main.flag\_module(module\_id: int)**

Flag saved module as whitelisted or blacklisted, depending on request.args.get('whitelisted')

**main.get\_blacklist()**

**main.get\_modules(whitelisted: bool)**

Get modules saved in the database, either blacklisted or whitelisted, as JSON response

**main.get\_searchterms()**

get all search terms from DB

**main.get\_studyprograms()**

Get distinct studyprograms associated with modules in the whitelist

**main.get\_studyprograms\_modules()**

Get Module-Studyprogramids associations as a dictionary

**main.get\_whitelist()**

**main.hello\_world()**

Hello World test view

**main.info()**

Information about the API

**main.public()**

Public front end view

**main.remove\_module(module\_id: int)**

remove module from database by id

**main.remove\_searchterm**(*searchterm\_id: int*)

remove searchterm from DB via id

**main.require\_appkey**(*view\_function*)

decorator for checking the api-key, making unauthorized requests impossible

**main.save\_studyprograms\_for\_module**(*module\_id: int, studyprograms: list*)

Save studyprogams for module in database, establish relationship

**main.search**()

get modules based on search terms, marking those already on white- and blacklist

**main.search\_upwards**()

Find course matches, then find containing modules, # and containing study programs

**main.update**()

Update saved modules to match their course catalogue counterparts, be there any changes

**main.wrap\_execute\_for\_modules\_in\_course**(*course*)

Wrapper function to be able to parallelize finding studyprograms for modules