

# Model

## Entity

```
§ User
  § id
  § email
  § password
  § urls
    + constructor // to create collection users if not exists
    + getters and setters
    //+ addUrl(url) = push
    //+ removeUrl(url) = splice

§ Url
  § id
  § url
  § urlMinified
  § userId
    + getters and setters
```

## Repository


- UserRepository

```
§ insertUser(email, password)
§ findUserByEmail(email)
§ findUserById(id)
```
- UrlRepository

```
§ findAllUserUrls(user) = user.urls
§ findUrlByUrl(url, user) // user or UserId?
§ insertUrl(url, userId) // updateUser : addUrl(url)
§ deleteUrl(url, userId)
```

## View

Registration page: .../register

 Error message X


email

\*\*\*\*\*

\*\*\*\*\*

OK

Login page: .../connect

 Error message X

email

\*\*\*\*\*

OK

Home page: .../get-urls

You are registered as user.email

url

OK

/add-url

ID	url	url_minified	
<i>for url in user.urls</i>			
1	http://google.fr	....	delete
2 = <i>url.ic</i>	<i>url.ur</i>	<i>url.urlMinified</i>	<i>/remove-url/ic</i>
3			

## Controller

- UserController
  - *emailValidator()* - TU
  - *passwordValidator()* - TU
  - *passwordConfirm()* - TU
  - *passwordHash()* - TU
  - *saveUser(user)* - TU //localStorage.setItem('user', user) or app.locals.user=user
  - *getUser()* - TU //localStorage.getItem('user')
  - *registerAction(req, res)* - TI // get for view .../register
  - *signUpAction(req, res)* - TI // post for action .../register
  - *connectAction(req, res)* - TI // get for view .../connect
  - *loginAction(req, res)* - TI // post for action .../connect
  - *logoutAction(req, res)* - TI
  - *indexAction(req, res)* - TI // get for home view .../
- UrlController
  - *urlValidator(url)* - TU
  - *idValidator(id)* - TU
  - *minifyUrl(url)* - TU
  - *addAction(req, res)* - TI // post .../add-url
  - *removeAction(req, res)* - TI // get .../remove-url/:id(\d+)
  - *getUrlsAction(req, res)* - TI // get .../get-urls

### Example:

#### *urlRouter.post('/add-url', urlController.addAction)*

*addAction():* *getForm()*, *getUserInput()*, *urlValidator(url)*, *minifyUrl(url)*, *getUser()*, *insertUrl(url, userId)*

#### *addAction(req, res){*

*//getForm - TU*

*let post = req.body;*

*//getUserInput = post.url - TU*

*if (urlValidator(post.url)){ // TU*

*let url = new Url;*

*//setUrl - TU*

*url.setUrl(post.url);*

*// minifyUrl(url) - TU and setUrlMinified(url) - TU*

*url.setUrlMinified( this.minifyUrl(post.url) ); // ? this = userController*

```

        //getUser - TU getId - TU
        let userId = app.locals.user.getId();

        //insert into database
        urlRepository.insertUrl(url, userId);
    }
}

```

/\*\*\*\*\*

*The same with JS*

```

//getForm(); - TU
document.getElementById('add-url-form').addEventListener('submit', addUrl);
function addUrl() { // TF
    //getUserInput(); - TU
    const url = document.getElementById('add-url-input').value;
    if (urlValidator(url)) { // TU
        //getUser() - TU
        const user = localStorage.getItem('user'); // TU
        //insert into database (syntax to verify): insertUrl(url, userId):
        db.users.update({id: 'user.id'},
            {$push: {urls: {url: url, urlMinified: minifyUrl(url)}}});
    }
}
}

```

/\*\*\*\*\*

*For TF it would be good to verify if you logout and login whether the user and list of urls are changed*