Documentation: CA 5 FINAL

5.12.2014

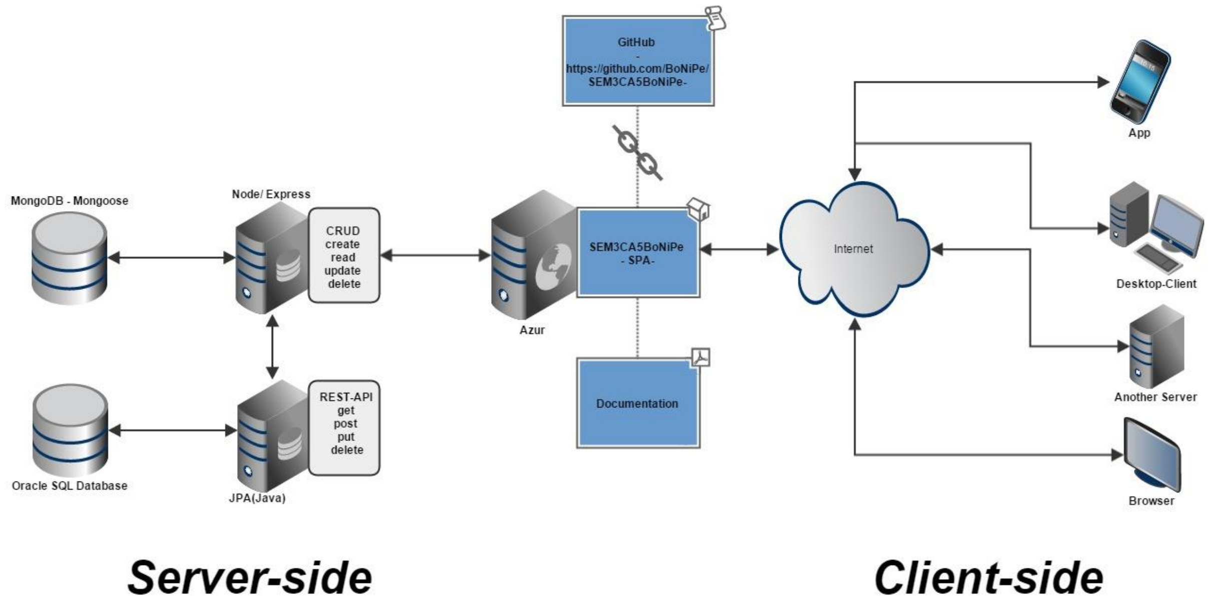
Names:

Boyko Surlev

Nikolaj Desting

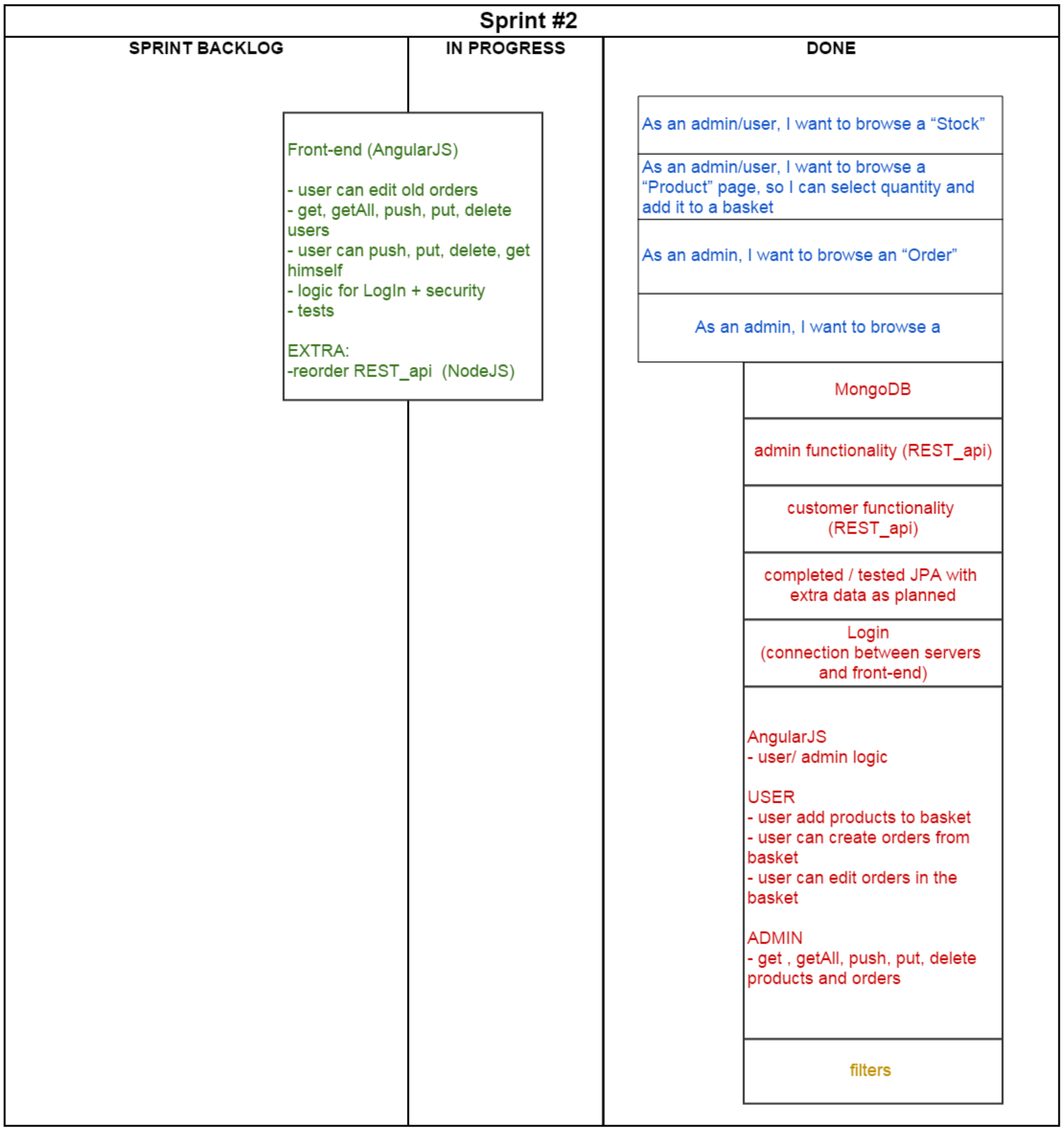
Peter Tomascik

Architecture diagram



http://tomascikpeter.wix.com/sem3ca5final ←----- **UI mock-up**

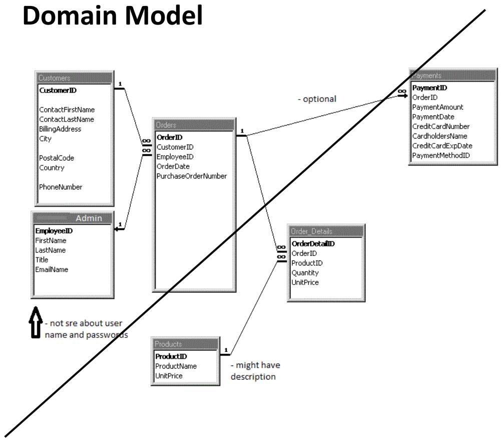
Sprint #2

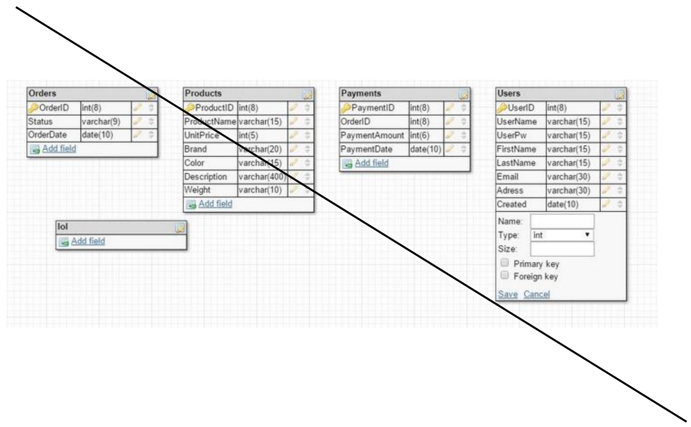


Sprint #2

SPRINT BACKLOG	IN PROGRESS	DONE
	Front-end (AngularJS) - user can edit old orders - get, getAll, push, put, delete users - user can push, put, delete, get himself - logic for LogIn + security - tests EXTRA: -reorder REST_api (NodeJS)	As an admin/user, I want to browse a “Stock”
		As an admin/user, I want to browse a “Product” page, so I can select quantity and add it to a basket
		As an admin, I want to browse an “Order”
		As an admin, I want to browse a
		MongoDB
		admin functionality (REST_api)
		customer functionality (REST_api)
		completed / tested JPA with extra data as planned
		Login (connection between servers and front-end)
		AngularJS - user/ admin logic USER - user add products to basket - user can create orders from basket - user can edit orders in the basket ADMIN - get , getAll, push, put, delete products and orders
		filters

Domain Model





Orders	
OrderID	int(8)
Status	varchar(9)
OrderDate	date(10)

Products	
ProductID	int(8)
ProductName	varchar(15)
UnitPrice	int(5)
Brand	varchar(20)
Color	varchar(15)
Description	varchar(400)
Weight	varchar(10)

Payments	
PaymentID	int(8)
OrderID	int(8)
PaymentAmount	int(6)
PaymentDate	date(10)

Users	
UserID	int(8)
UserName	varchar(15)
UserPw	varchar(15)
FirstName	varchar(15)
LastName	varchar(15)
Email	varchar(30)
Adress	varchar(30)
Created	date(10)

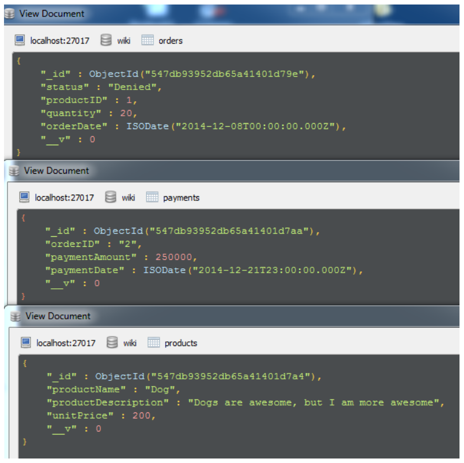
Name:
Type: int
Size:
Primary key
Foreign key

lol

JPA

ID	ADRESS	FNAME	LNAME	PASSWORD	TYPE	USERALIAS	USERNAME		
1	444	dfg	priezv nic	5906ac361a137e2d286465cd6588ebb5ac3f5ae9550011	admin	holal	97c7403d9bf0573235766d0f6fdeff94b3a261		
2	Bulgaria	Peter	Sakula	6cf615d5bcaac778352a8f1f3360d23f02f34ec182e25989	customer	dragon	9d104eaf685780a2159f11460d2e8e0adc8b3		
5	TRY	TRY	TRY	fd237ee90f0e2d65078bca2766f7727bc6510a151c5fc50	customer	TRY	fd237ee90f0e2d65078bca2766f7727bc6510		

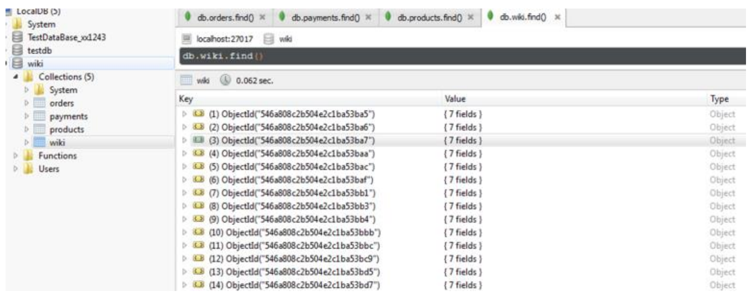
MONGO DB



```
{
    "_id" : ObjectId("547db93952db65a41401d79e"),
    "status" : "Denied",
    "productID" : 1,
    "quantity" : 20,
    "orderDate" : ISODate("2014-12-08T00:00:00.000Z"),
    "__v" : 0
}
```

```
{
    "_id" : ObjectId("547db93952db65a41401d7aa"),
    "orderID" : "2",
    "paymentAmount" : 250000,
    "paymentDate" : ISODate("2014-12-21T23:00:00.000Z"),
    "__v" : 0
}
```

```
{
    "_id" : ObjectId("547db93952db65a41401d7a4"),
    "productName" : "Dog",
    "productDescription" : "Dogs are awesome, but I am more awesome",
    "unitPrice" : 200,
    "__v" : 0
}
```



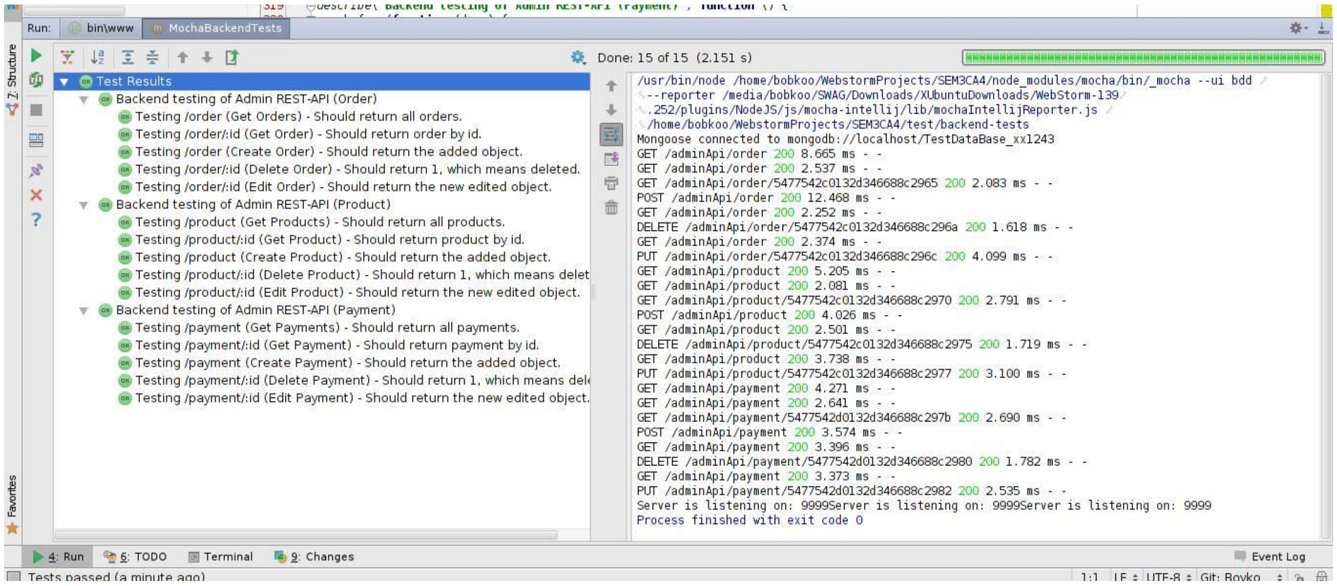
```
db.wiki.find()
```

wiki 0.062 sec.

Key	Value	Type
(1) ObjectId("546a808c2b504e2c1ba53ba5")	{ 7 fields }	Object
(2) ObjectId("546a808c2b504e2c1ba53ba6")	{ 7 fields }	Object
(3) ObjectId("546a808c2b504e2c1ba53ba7")	{ 7 fields }	Object
(4) ObjectId("546a808c2b504e2c1ba53baa")	{ 7 fields }	Object
(5) ObjectId("546a808c2b504e2c1ba53bac")	{ 7 fields }	Object
(6) ObjectId("546a808c2b504e2c1ba53baf")	{ 7 fields }	Object
(7) ObjectId("546a808c2b504e2c1ba53bb1")	{ 7 fields }	Object
(8) ObjectId("546a808c2b504e2c1ba53bb3")	{ 7 fields }	Object
(9) ObjectId("546a808c2b504e2c1ba53bb4")	{ 7 fields }	Object
(10) ObjectId("546a808c2b504e2c1ba53bbb")	{ 7 fields }	Object
(11) ObjectId("546a808c2b504e2c1ba53bbc")	{ 7 fields }	Object
(12) ObjectId("546a808c2b504e2c1ba53bc9")	{ 7 fields }	Object
(13) ObjectId("546a808c2b504e2c1ba53bd5")	{ 7 fields }	Object
(14) ObjectId("546a808c2b504e2c1ba53bd7")	{ 7 fields }	Object

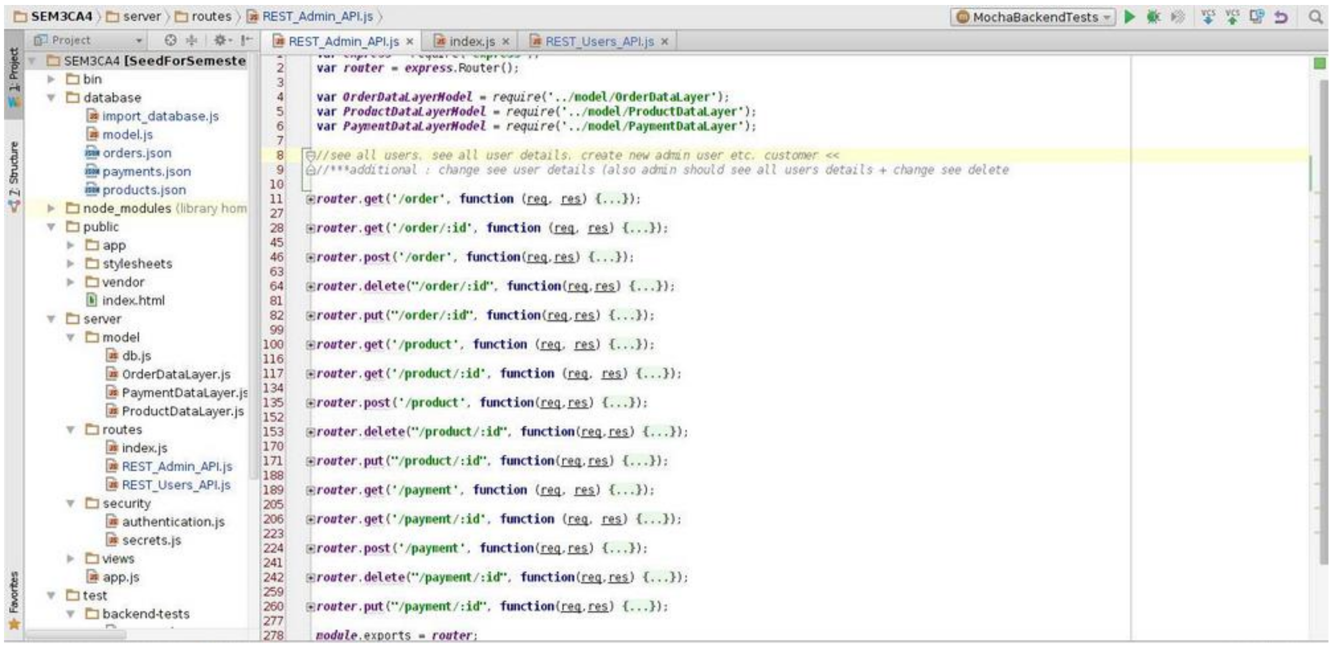
Back-end

Rest_API (Node) + Mocha tests



```
Test Results
  Backend testing of Admin REST-API (Order)
    Testing /order (Get Orders) - Should return all orders.
    Testing /order/:id (Get Order) - Should return order by id.
    Testing /order (Create Order) - Should return the added object.
    Testing /order/:id (Delete Order) - Should return 1, which means deleted.
    Testing /order/:id (Edit Order) - Should return the new edited object.
  Backend testing of Admin REST-API (Product)
    Testing /product (Get Products) - Should return all products.
    Testing /product/:id (Get Product) - Should return product by id.
    Testing /product (Create Product) - Should return the added object.
    Testing /product/:id (Delete Product) - Should return 1, which means delet
    Testing /product/:id (Edit Product) - Should return the new edited object.
  Backend testing of Admin REST-API (Payment)
    Testing /payment (Get Payments) - Should return all payments.
    Testing /payment/:id (Get Payment) - Should return payment by id.
    Testing /payment (Create Payment) - Should return the added object.
    Testing /payment/:id (Delete Payment) - Should return 1, which means dele
    Testing /payment/:id (Edit Payment) - Should return the new edited object.
```

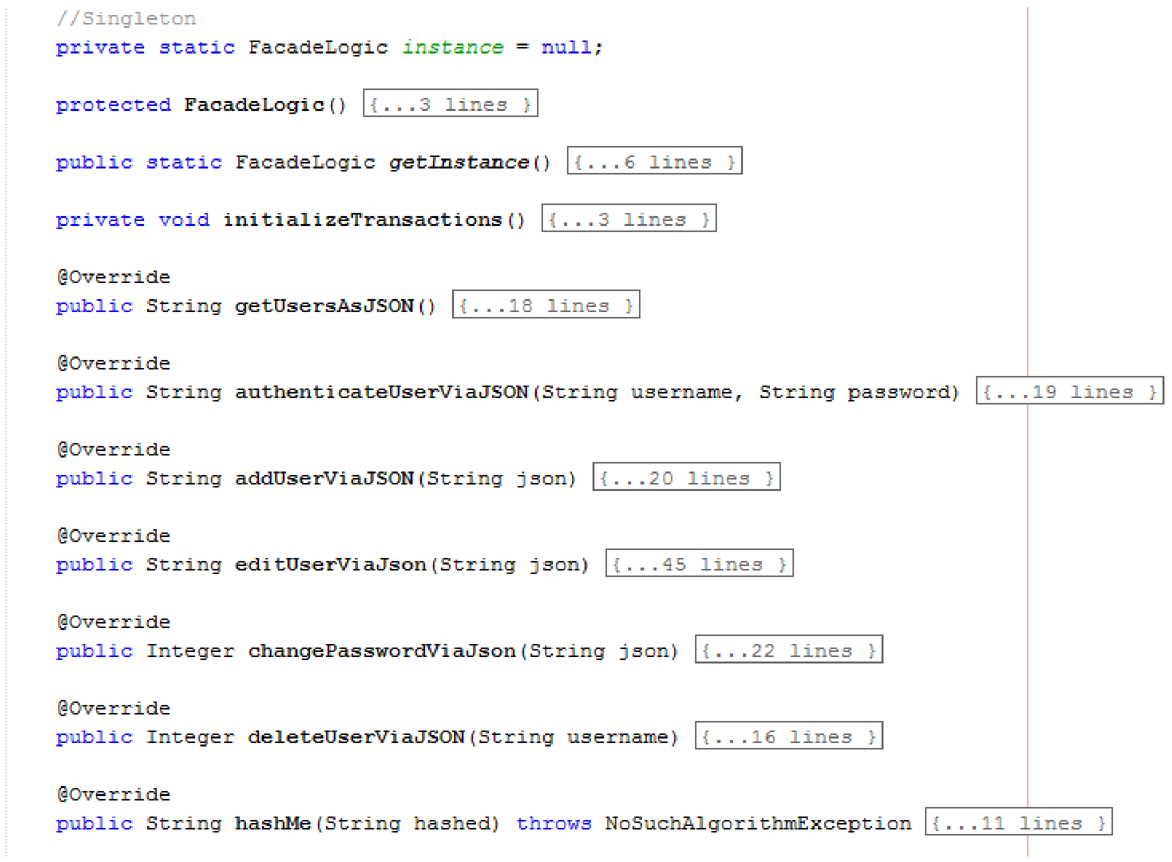
```
/usr/bin/node /home/bobkoo/WebstormProjects/SEM3CA4/node_modules/mocha/bin/_mocha --ui bdd
 --reporter /media/bobkoo/SWAG/Downloads/XUbuntuDownloads/WebStorm-139
 .252/plugins/NodeJS/js/mocha-intellij/lib/mochaIntellijReporter.js
 /home/bobkoo/WebstormProjects/SEM3CA4/test/backend-tests
Mongoose connected to mongodb://localhost/TestDataBase_xx1243
GET /adminApi/order 200 8.665 ms - -
GET /adminApi/order 200 2.537 ms - -
GET /adminApi/order/5477542c0132d346688c2965 200 2.083 ms - -
POST /adminApi/order 200 12.468 ms - -
GET /adminApi/order 200 2.252 ms - -
DELETE /adminApi/order/5477542c0132d346688c296a 200 1.618 ms - -
GET /adminApi/order 200 2.374 ms - -
PUT /adminApi/order/5477542c0132d346688c296c 200 4.099 ms - -
GET /adminApi/product 200 5.205 ms - -
GET /adminApi/product 200 2.081 ms - -
GET /adminApi/product/5477542c0132d346688c2970 200 2.791 ms - -
POST /adminApi/product 200 4.026 ms - -
GET /adminApi/product 200 2.501 ms - -
DELETE /adminApi/product/5477542c0132d346688c2975 200 1.719 ms - -
GET /adminApi/product 200 3.738 ms - -
PUT /adminApi/product/5477542c0132d346688c2977 200 3.100 ms - -
GET /adminApi/payment 200 4.271 ms - -
GET /adminApi/payment 200 2.641 ms - -
GET /adminApi/payment/5477542d0132d346688c297b 200 2.690 ms - -
POST /adminApi/payment 200 3.574 ms - -
GET /adminApi/payment 200 3.396 ms - -
DELETE /adminApi/payment/5477542d0132d346688c2980 200 1.782 ms - -
GET /adminApi/payment 200 3.373 ms - -
PUT /adminApi/payment/5477542d0132d346688c2982 200 2.535 ms - -
Server is listening on: 9999Server is listening on: 9999Server is listening on: 9999
Process finished with exit code 0
```



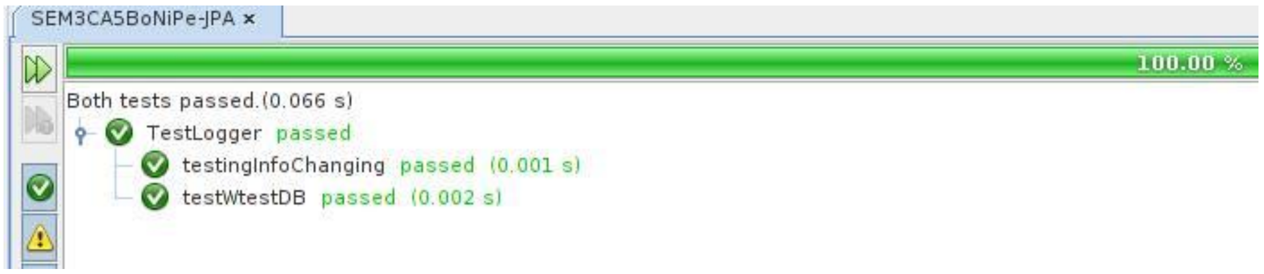
```
SEM3CA4 [SeedForSemeste
  bin
  database
    import_database.js
    model.js
    orders.json
    payments.json
    products.json
  node_modules (library hom
  public
    app
    stylesheets
    vendor
    index.html
  server
    model
      db.js
      OrderDataLayer.js
      PaymentDataLayer.js
      ProductDataLayer.js
    routes
      index.js
      REST_Admin_API.js
      REST_Users_API.js
    security
      authentication.js
      secrets.js
    views
    app.js
  test
    backend-tests
```

```
var router = express.Router();

var OrderDataLayerModel = require('../model/OrderDataLayer');
var ProductDataLayerModel = require('../model/ProductDataLayer');
var PaymentDataLayerModel = require('../model/PaymentDataLayer');

//see all users. see all user details. create new admin user etc. customer <<
//***additional : change see user details (also admin should see all users details + change see delete

router.get('/order', function (req, res) {...});

router.get('/order/:id', function (req, res) {...});

router.post('/order', function(req,res) {...});

router.delete("/order/:id", function(req,res) {...});

router.put("/order/:id", function(req,res) {...});

router.get('/product', function (req, res) {...});

router.get('/product/:id', function (req, res) {...});

router.post('/product', function(req,res) {...});

router.delete("/product/:id", function(req,res) {...});

router.put("/product/:id", function(req,res) {...});

router.get('/payment', function (req, res) {...});

router.get('/payment/:id', function (req, res) {...});

router.post('/payment', function(req,res) {...});

router.delete("/payment/:id", function(req,res) {...});

router.put("/payment/:id", function(req,res) {...});

module.exports = router;
```

JAVA + J-Unit



```
//Singleton
private static FacadeLogic instance = null;

protected FacadeLogic() {...3 lines }

public static FacadeLogic getInstance() {...6 lines }

private void initializeTransactions() {...3 lines }

@Override
public String getUsersAsJSON() {...18 lines }

@Override
public String authenticateUserViaJSON(String username, String password) {...19 lines }

@Override
public String addUserViaJSON(String json) {...20 lines }

@Override
public String editUserViaJson(String json) {...45 lines }

@Override
public Integer changePasswordViaJson(String json) {...22 lines }

@Override
public Integer deleteUserViaJSON(String username) {...16 lines }

@Override
public String hashMe(String hashed) throws NoSuchAlgorithmException {...11 lines }
```



```
SEM3CA5BoNiPe-JPA ×
100.00 %
Both tests passed.(0.066 s)
  TestLogger passed
    testingInfoChanging passed (0.001 s)
    testWtestDB passed (0.002 s)
```

Who did what?

What?	Who?
Java + JPA	Peter (help of Boyko)
Node/Express + MongoDb	Boyko
Tests(all)	Boyko
documentation	Peter
backlog	Nick
sprint backlog	Peter
architecture diagram	Peter
deploying	Nick
Domain model	Nick+Peter
UI mock-ups	Nick+Peter