|  |  |  |
| --- | --- | --- |
| **Description** | **Method + Path**  path is alwayshttp://localhost:8085/ | **Request/Response** |
| Get Store Statistics (displayed on home page - right hand side) | GET | **Request** **{**store**}**    **Response {**  status:”ok”,  content {"**numberProducts**":70,"**numberOrders**":45}  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Check if user (customer or administrator) Logged in  Need to know what to display on home page  If logged in user found  Response object contains **customerInfo** object | GET | **Request {**login**}**  **Response** (no logged in user found) **{**  status: "ok",  content: {**cookie**:{**originalMaxAge**: null, **expires**: null, **httpOnly**: true,  **path**: "/"}  **}**  **Response** (logged in user found) **{**  status: "ok",  content:{"**cookie**":{"**originalMaxAge**":null,"**expires**":null, "**httpOnly**":true,"**path**":"/"},  "**customerInfo**":**{**"**teudatZehut**":1,  "**firstName**":"Donald",  "**lastName**":"Trump",  "**email**":"1",  "**password**":"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*",  "**street**":"Brookside Road 32",  "**city**":"Houston",  "**role**":"customer"**}**}  **}** |
| Login – send login data (**loginInfo** object) to server.  If customer/admin found with **loginInfo** save customer/admin data in  **customerInfo** object in server session. | POST | **Request** **{**login,  **loginInfo:** {"email":"1@gmail.com","passWord":"1"}  **}**  **Response** (customer/admin found with login data) **{**  status:"ok",  content: {"**teudatZehut**":313646010,  "**firstName**":"Judith",  "**lastName**":"Ilson",  "**email**":"judithguttmann@gmail.com",  "**password**":"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*",  "**street**":"Hakablan 69 apt # 2",  "**city**":"Jerusalem",  "**role**":"admin"}  **}**  **Response** (no customer/admin found with login data) **{**  status:"noSuchCustomer", content:""  **}**  **Response** (error occurred on server – entry written to error log) **{**  status: "error", content: error message  **}** |
| Logout – destroy customerInfo session object on server | DELETE | **Request** **{**login**}**  **Response {**  status: "ok", content: ""  **}** |
| Get all categories – used to fill combo box in product update panel/controller | GET | **Request** **{**category**}**  **Response {**  **{**status: "ok",  content: {value: 1, text: "Bakery"},  {value: 2, text: "Chocolates & Sweets"},  {value: 3, text: "Vegetables & Fruits"} … **]}**  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Get all products  Request source parameter  used to determine what kind of illegal access to product data was attempted if person accessing product data is not logged in (i.e no **customerInfo** object found in session on server) | GET | **Request {** product,  **Source=**/products **or Source=**/shop  **}**  **Response {**  status:"ok",  content:**[**  {"**id**":1,"**name**":"diet coke can","**category**":18,"**price**":0.49},  {"**id**":2,"**name**":"Doritos","**category**":19,"**price**":0.89},  {"**id**":3,"**name**":"dried apricots","**category**":16,"**price**":13.95},  {"**id**":4,"**name**":"eggplant","**category**":16,"**price**":6.75},  {"**id**":5,"**name**":"eggs","**category**":15,"**price**":4.99}**]**  **}**  **Response** (attempt to access product data without being logged in) **{**  status:"forbiddenAccessAttempted", content:""  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Create new product | POST | **Request {** product,  **id**: value-from-client-is-undefined,  **name**: pizza,  **category**: 1,  **price**: 6.99,  **productImage**; filename="38.png"  **}**  **Response** (product added successfully) **{**  status:"ok",  content:"product added successfully => new productID: 94"  **}**  **Response** (product data sent to server invalid) **{**  status: "invalid input ",  content: " invalid input => following errors were found: "  product price up to 9999.99 $ required,  product image file extension invalid - valid extensions are: jpg, jpeg,  png or gif  **}**  **Response** (attempt to insert product without being logged in or admin) **{**  status:"forbiddenAccessAttempted",content:""  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error", content: error message  **}** |
| Update existing product | PUT | **Request {** product,  **id**: 94,  **name**: pizza with olives,  **category**: 1,  **price**: 5.99,  **productImage**; filename="38.png"  **}**  **Response** (product updated successfully) **{**  status:"ok",content: "product updated successfully"  **}**  **Response** (product data sent to server invalid) **{**  status: "invalid input",  content:"invalid input => following errors were found:  product category required,  product image larger than 5MB - actual size: 5123456 bytes  **}**  **Response** (attempt to insert product without being logged in or admin) **{**  status:"forbiddenAccessAttempted", content:""  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Get last cart info for given customer – last cart = cart with most recent **creationDate** | GET | **Request**{ cart,  **teudatZehut**=1  }  **Response {**  status:"ok",  content: {"**id**":68,  "**customer**":1,  "**creationDate**":"2018-03-24T21:17:30.000Z",  "**orderDate**":""}  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Create cart for customer with no open cart.  Open cart = cart for which no order was submitted **orderDate** = “”.  On successful cart creation id of new cart is returned in content object of server response | POST | **Request**{ cart,  **teudatZehut**=5  }  **Response {**  status:"ok",  content: **{**"70”**}**  **}**  **Response** (user not logged in or customer attempting to access order  panel directly (without going through shopping cart)) **{**  status:"forbiddenAccessAttempted", content:""  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| Get cart items (products) for given cart ID . | GET | **Request**{ cartItem,  **cartID**=70  }  **Response {**  status:"ok",  content: **[**{"**id**":208,  "**productID**":81,  "**productName**":"french croissant"  ,"**productPrice**":0.99,  "**quantity**":1,  "**price**":1.99,  "**shoppingCart**":52},  {"id":227,"productID":31,"productName":"red wine","productPrice":26.99,"quantity":2,"price":53.98,"shoppingCart":52},{"id":228,"productID":83,"productName":"farmer's bread","productPrice":3.19,"quantity":1,"price":3.19,"shoppingCart":52}**]**  **}**  **Response** (error occurred on server – entry written to error log) **{**  status:"error",  content: error message  **}** |
| *Create new Course*  *duplicateCourse & courseImageSize*  *are used for client validations (they are NOT required on the server)*  *They can be found in Request because they are in form and whole form is serialized with the*  *$("#frmCUD").serialize();*  *Statement.* | *POST* | ***Request****{*  ***ctrl:*** *course,*  ***courseID****: 0,*  ***duplicateCourse****:,*  ***courseName****: karate,*  ***courseDescription****: defend yourself,*  ***courseImageSize****:,*  ***courseImage****: ; filename="002.JPG"*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":"Create course",*  *"message":" course added successfully",*  *"new\_courseID":15*  *}* |
| *Update Course*  *duplicateCourse & courseImageSize*  *are used for client validations (they are NOT required on the server)*  *They can be found in Request because they are in form and whole form is serialized with the*  *$("#frmCUD").serialize();*  *Statement.* | *PUT* | ***Request****{*  ***ctrl:*** *course,*  ***courseID****: 3,*  ***duplicateCourse****:,*  ***courseName****: c# - beginners,*  ***courseDescription****: learn to program- fast,*  ***courseImageSize****:*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":" Update course",*  *"message":" course updated successfully"*  *}* |
| *Delete Course* | *DELETE* | ***Request****{*  ***ctrl:*** *course,*  ***courseID****: 3*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":" Delete course ",*  *"message":" course deleted successfully"*  *}* |
| *Get student by name, phone & mail – used for remote validation on client to ensure no other student with same name, phone & mail already exists*  *studentID: -1 no such student found*  *studentID: <> -1 such student found* | *GET* | ***Request****{*  ***ctrl:*** *student,*  ***studentName****:* *Efraim Frei,*  ***studentPhone:*** *052 715 6769****,***  ***studentEmail:*** *judy@gmail.com*  *}*  ***Response {***  *{"status":"ok",*  *" studentID":17}*  *}*  ***Response*** *(no session variable on server - user no longer logged in)* ***{***  *{"status":"no longer logged in",*  *"action":"GET student"}*  *}* |
| *Get all students* | *GET* | ***Request****{*  ***ctrl:*** *student*  *}*  ***Response {***  *[{"studentID":7,*  *"studentName":"Ming-Yang Xie",*  *"studentPhone":"02 - 651 3568",*  *"studentEmail":"judithguttmann@gmail.com",*  *"studentCourses":"cbCourse1,cbCourse2"},*  *{"studentID":3,*  *"studentName":"Thomas Axen”,*  *"studentPhone":"559888705",*  *"studentEmail":"judithguttmann@gmail.com",*  *"studentCourses":"cbCourse2,cbCourse4,cbCourse5"} … ]*  *}* |
| *Create new Student*  *duplicateStudent & studentImageSize*  *are used for client validations (they are NOT required on the server)*  *They can be found in Request because they are in form and whole form is serialized with the*  *$("#frmCUD").serialize();*  *Statement.* | *POST* | ***Request****{*  ***ctrl:*** *student,*  ***studentID****: 0,*  ***duplicateStudent****:,*  ***studentName****: Shimi Bloch,*  ***studentPhone****: 09 – 835 9029,*  ***studentEmail****:* [*judy@gmail.com*](mailto:judy@gmail.com)*,*  ***studentImageSize****:,*  ***studentImage****: ; filename="013.JPG"*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":"Create student ",*  *"message":" student added successfully",*  *"new\_studentID":15*  *}* |
| *Update Student*  *duplicateStudent & studentImageSize*  *are used for client validations (they are NOT required on the server)*  *They can be found in Request because they are in form and whole form is serialized with the*  *$("#frmCUD").serialize();*  *Statement.* | *PUT* | ***Request****{*  ***ctrl:*** *student,*  ***studentID****: 19,*  ***duplicateStudent****:,*  ***studentName****: Shimi Bloch,*  ***studentPhone****: 09 – 835 9029 grandma’s phone,*  ***studentEmail****:* [*judy@gmail.com*](mailto:judy@gmail.com)*,*  ***studentImageSize****:*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":"Update student ",*  *"message":" student updated successfully"*  *}* |
| *Delete Student* | *DELETE* | ***Request****{*  ***ctrl:*** *student,*  ***studentID****: 3*  *}*  ***Response*** *{*  *"status":"ok",*  *"action":" Delete student",*  *"message":" student deleted successfully"*  *}* |