NILE BOOKS

MILES MORE THAN AMAZON

Bigger Than The Amazon™

Final Project (4) Submission: Web Services Programming

Max Claypool, Michael Flatley, J. Curtis Main

Concept

- Nile Books allows users to partner with one service to post and sell products that will be available at one convenient location.
  - We provide a database for the partners to store their data, and tools to manipulate their items in the database with just a few simple clicks.
- Partner users will be able to sign in, add products, and manage their stock.
- Customer users will be able to sign in, view products, and manage their orders.

Services

- Entry points with search and for customers
- Manipulation via links/HATEOAS
- Authentication/authorization for customers/partners
- Product purchase
- Status and cancelation
- Resources can be listed

Services

- Entry points with search and for customers
- Manipulation via links/HATEOAS
- Authentication/authorization for customers/partners
- Product purchase
- Status and cancelation
- Resources can be listed

API Functions

Exposed Customer Methods

//customer functions

- public CustomerRep getCustomer(CustomerRequest request);
- public CustomerRep updateCustomer(CustomerRequest request);
- public void deleteCustomer(CustomerRequest request);
- public CustomerRep createCustomer(CustomerRequest request);

//product functions for customers

- public List<ProductRep> getAllProducts();
- public List<ProductRep> getProductsByTitle(String title);
- public List<ProductRep> getProductsByGenre(String genre);
- public List<ProductRep> getProductsByAuthor(String author);

Exposed Customer Methods

//order functions for customer

- public List<OrderRep> getOrdersByAttribute(String attribute, String value);
- public List<OrderRep> getAllOrders(String attribute, String value);
- public List<OrderRep> getShoppingCart(String attribute, String value);
- public List<OrderRep> createShoppingCart(String attribute, String value);
- public List<OrderRep> addItemToShoppingCart(OrderRequest request);
- public List<OrderRep> placeOrder(int id);
- public void orderCancellationRequest(int id);
- public void returnOrderRequest(int id);

Exposed Partner Methods

//Partner functions

- public PartnerRep getPartner(PartnerRequest request);
- public PartnerRep updatePartner(PartnerRequest request);
- public void deletePartner(PartnerRequest request);
- public PartnerRep createPartner(PartnerRequest request);

//partner methods for order

- public List<OrderRep> getOrdersByAttribute(String attribute, String value);
- public List<OrderRep> getAllOrders(String attribute, String value);
- public List<OrderRep> updateOrder(List<OrderRequest> request);
- public void deleteOrder(int id);
- public void cancelOrder(int id);

Exposed Partner Methods

//product methods for partner

- public List<ProductRep> getProductsByTitle(String title);
- public List<ProductRep> getProductsByGenre(String genre);
- public List<ProductRep> getProductsByAuthor(String author);
- public List<ProductRep> getAllProducts();
- public List<ProductRep> getProductsByPartnerId(String partnerID);
- public ProductRep createProduct(ProductRequest request);
- public void deleteProductById(int productId);
- public ProductRep updateProduct(ProductRequest request);

Exposed Product Methods

- public List<ProductRep> getProductsByTitle(String title);
- public List<ProductRep> getProductsByGenre(String genre);
- public List<ProductRep> getProductsByAuthor(String author);
- public List<ProductRep> getAllProducts();

Challenges & Considerations

Challenges (lots of them)

- We all work and have 3+ classes
  - Hard to coordinate meeting (mostly done on Google Hangouts)
  - Had to work on it a little at a time throughout the semester making it hard to get HATEOAS and the server working (spent too much time on both and fell behind on the website)
- We do not know Java and Eclipse at all
  - Took a long time to code
  - Couldn’t find the correct JARs quickly
  - Couldn’t get CRUD or HATEOAS to work on server
  - Problems getting Eclipse to work (installing JARs, Tomcat, CXF, creating projects, etc...)
  - Took a long time to get Hibernate and JBDC working
- Not the best with JSON and AJAX
  - Decided against using a server because of this

If We Extended the Project…

Database:

- Add “Product Review”
- Add payment for placing orders
- Encrypt payments and passwords

API:

- Add login functions
- Add product review functions
- Add payment to placing orders
- More functions in Order and Product
- Handling for DDoS and SQL injection
- Host API on a server with SSL

If We Extended the Project…

Website:

- Host on a server
- Add authentication prompt
- Build the website in Rails or Flask
- Create a server to fetch JSON from the DB
  - Current plan was to use JSON but API wasn’t working

API Demo

Website Demo

Questions?