Software Design and Development

TextForSale

Client

Arash Fallah

Team

Mehreen Awan
John Gordon
Daniel Schomisch
Daniel Kelly
Mina Beshai

TextForSale
System Design Document

Table of Contents

1. **Introduction**

1.1 Purpose of This Document

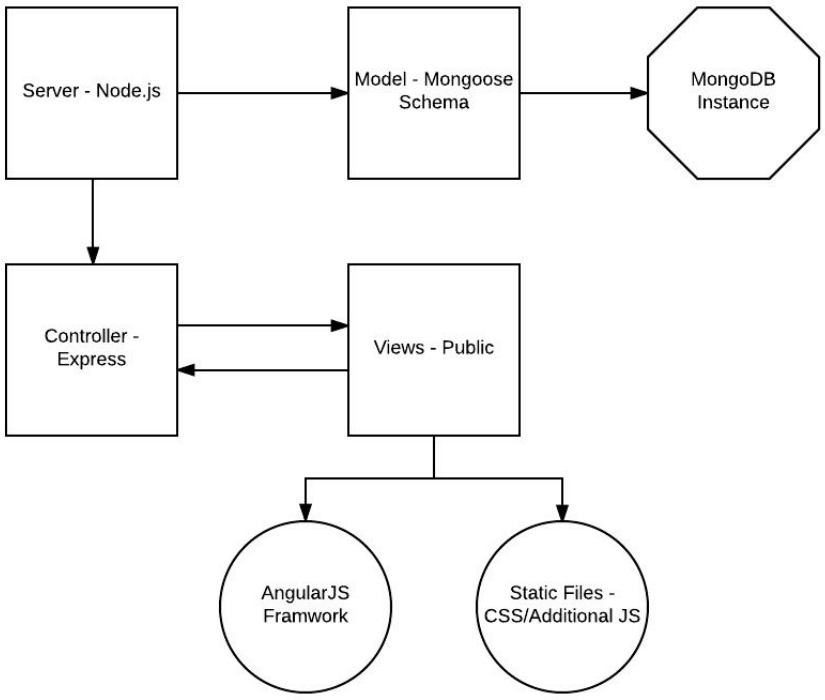
The purpose of this document is to describe the TextForSale application in detail. The following sections include a high-level detailed architectural design, component details and class design, and database schematic and management.

1.2 References

Mongoose - https://www.npmjs.com/package/mongoose
Two-Way Databinding - https://docs.angularjs.org/guide/databinding
NoSQL Database Implementation - https://www.mongodb.com/nosql-explained

2. **System Architecture**

2.1 Architectural Design



The TextForSale application will utilize four main components and several sub components, or Node packages. The four main components are MongoDB, Express, AngularJS, and Node.js, or the MEAN Stack.

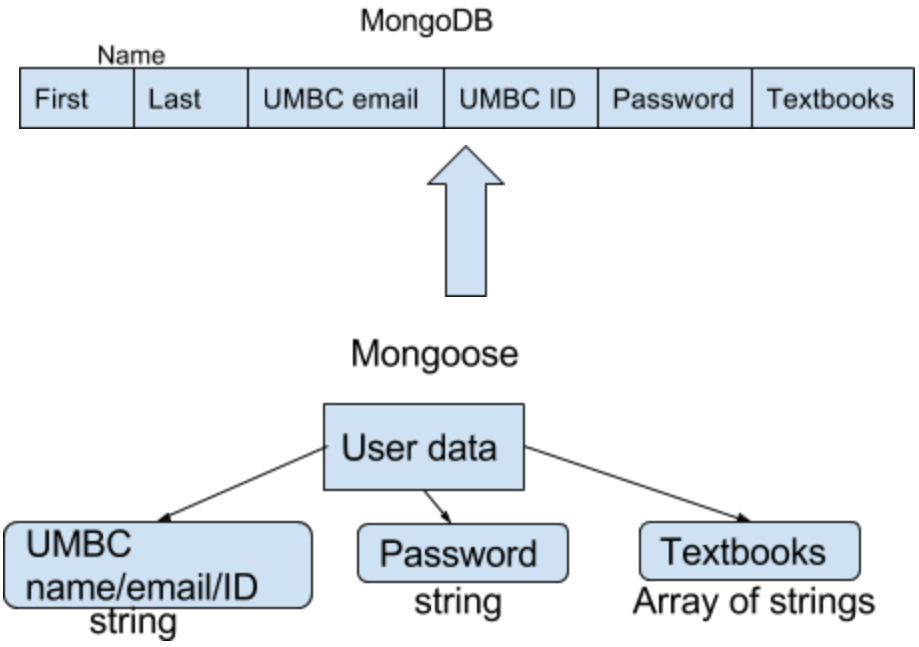
MongoDB is a NoSQL database which provides structured and unstructured collection of data. The TextForSale application will utilize a Node.js package named Mongoose to control the schema of structured data such as the ‘Users’ table. Mongoose can be viewed as a mediator between the data posted to the server and the MongoDB instance. Node.js is simply the server code written in Javascript. Node.js will interact with both the model and the controller to initiate the process of delivering data and content to the user. Express is the controller, or router, for the application. Express will provide REST characteristics for serving the persistent data of the application as well as controlling where users are allowed access. AngularJS is a front-end Javascript framework which will serve the static files and leverage more powerful experience for the user with features such as two-way data binding. Angular will handle embedding the persistent data within the static files, such as a list of popular textbooks on the home page. Any DOM manipulation done by the user will also update the model of the application.

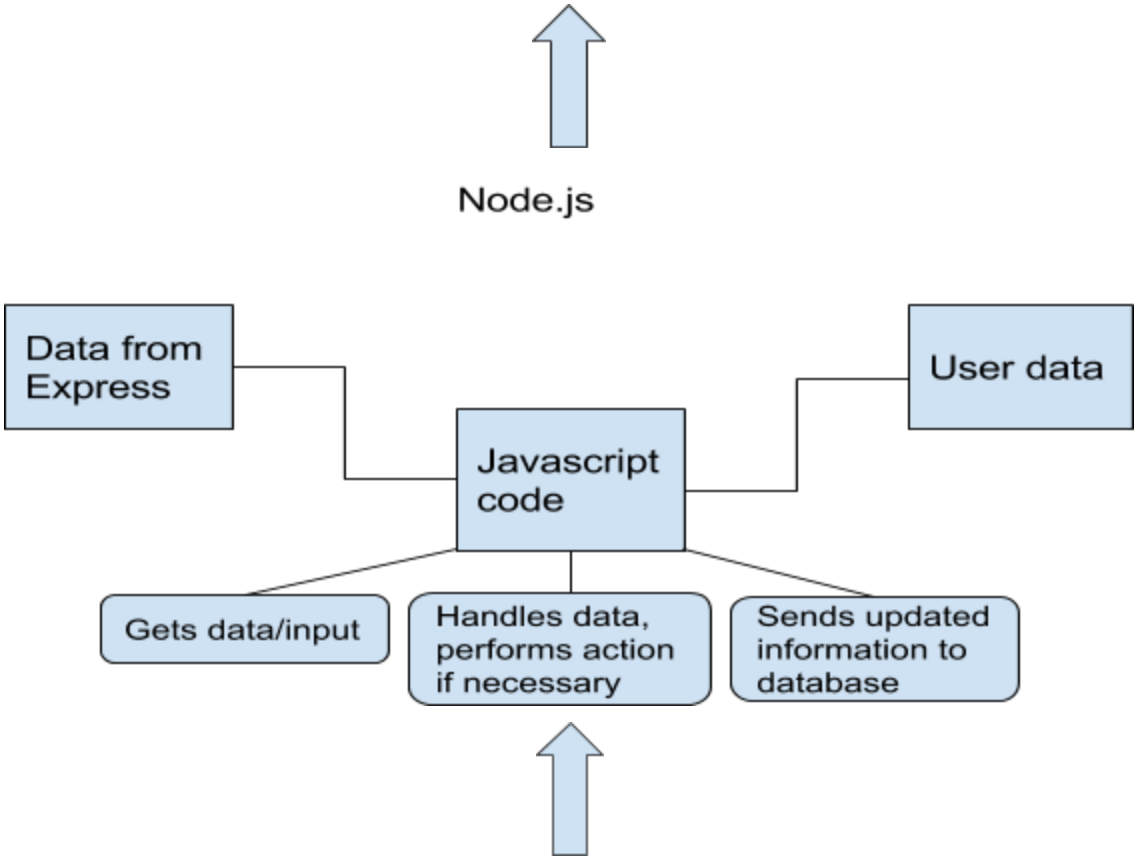
2.2 Decomposition Description

AngularJS handles user interaction, it is constantly checking for user input. It uses a packaged js (or multiple) to let the user input and then understands the raw input and sends it to the server (Node.js) to act on that input.

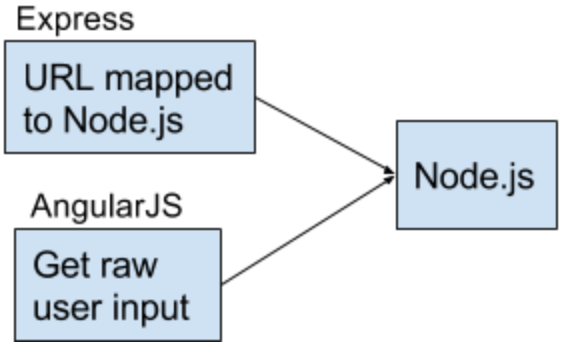
Express is mainly for mapping a url and sending any user input to the correct code in Node.js.

Mongoose is an instance of the database, MongoDB. In other words, Mongoose formats the data in a structured way for MongoDB to store.



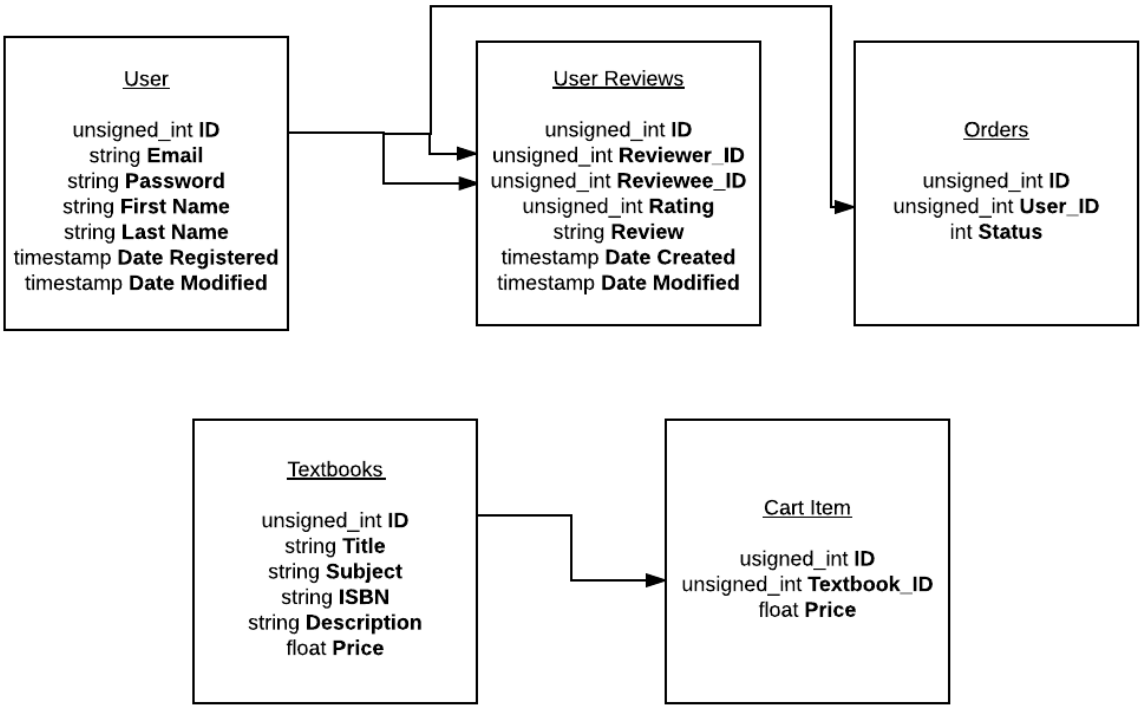


Express and AngularJS



3. **Persistent Data Design**

3.1 Database Descriptions



3.2 File Descriptions

The ‘User’ table will consist of an unique identifier, e-mail, password, first and last name, and dates for registration and information modification. The e-mail will be used both for registration confirmation and authorization to log in. The password will contain a hashed version of the user’s password to check against when logging in. The first name and last name will simply be used as an identifier and a formality for the front-end (e.g. a greetings page). The date registered and modified will provide detail about when the user was created to track the life span of the use of the application.

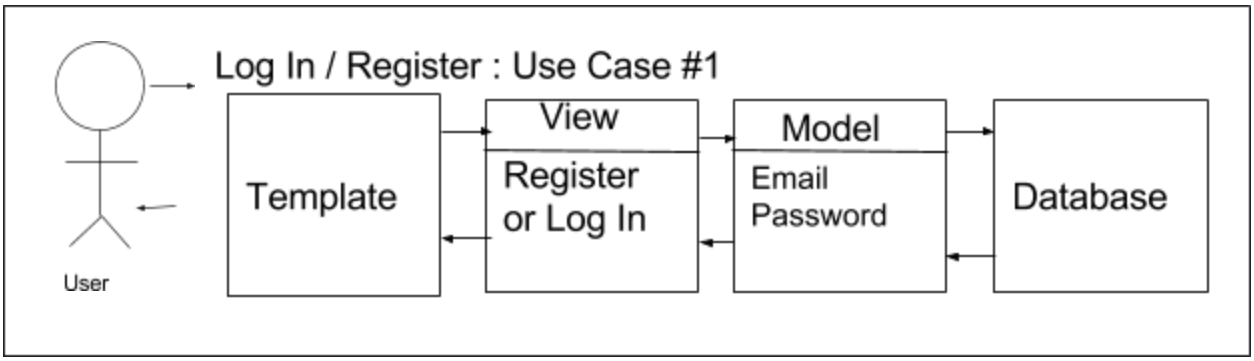
The ‘User Review’ table will contain a collection of data pertaining to user’s reviews. The table will contain a unique identifier, the user identifier of the reviewer, the user identifier of the reviewee, a numerical representation of the rating (e.g. 1-5 for stars), a plain text description of the actual review, and timestamps for creation and modification.

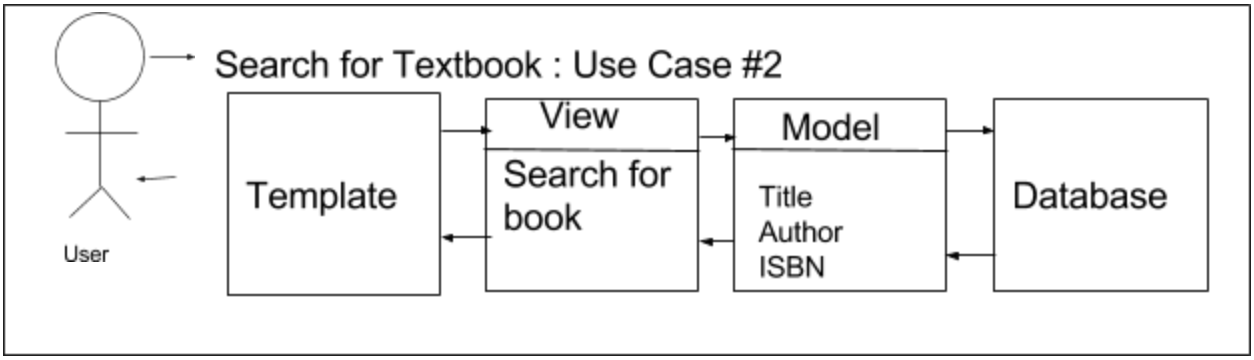
The "Orders" table will consist of a unique numerical identifier, a reference the user's id, and a status field to help identify if an order status returns paid/unpaid/error etc.

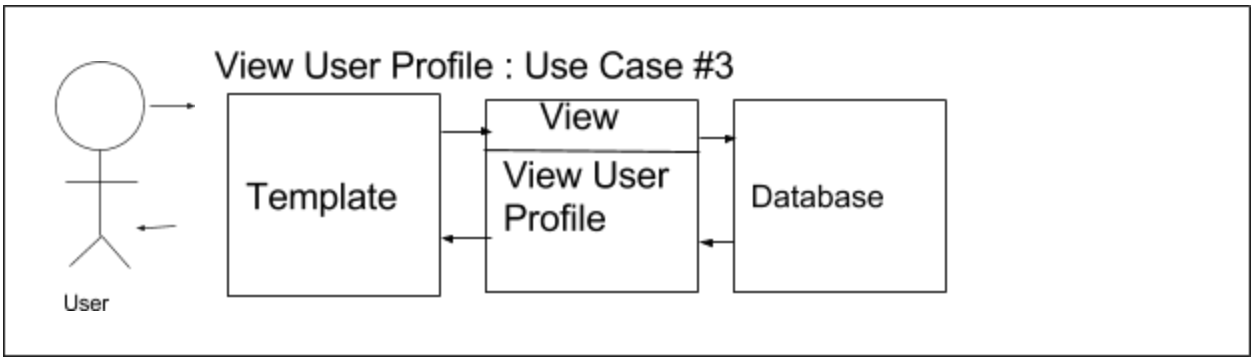
The "Textbooks" table will use a unique identifier, and have fields for the textbook's title, price, subject content, and a place to store a brief description about the book. ISBNs will also reside in this table to assist the search by ISBN feature.

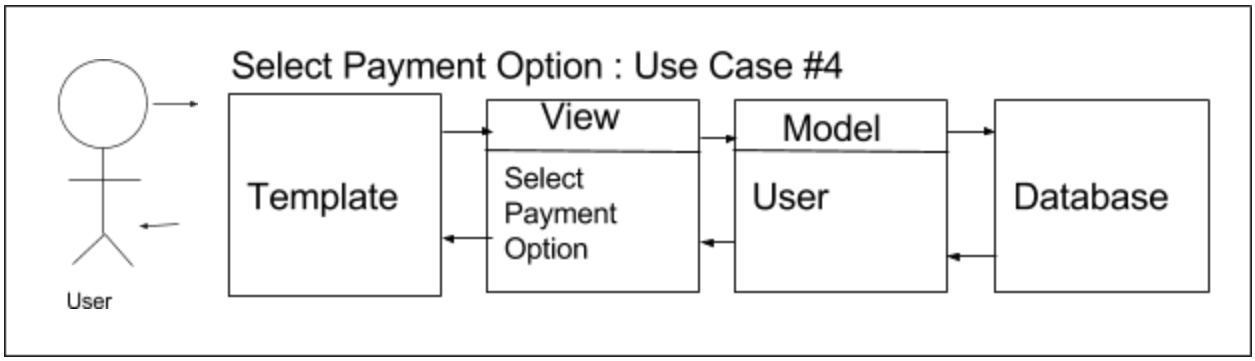
The "Cart Item" table has a unique identifier, price, and a reference to the textbook's unique id.

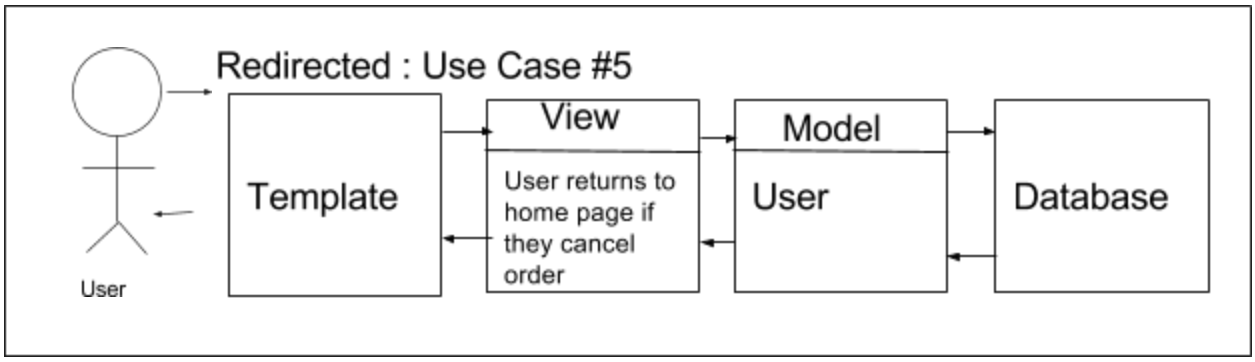
4. **Requirements Matrix**

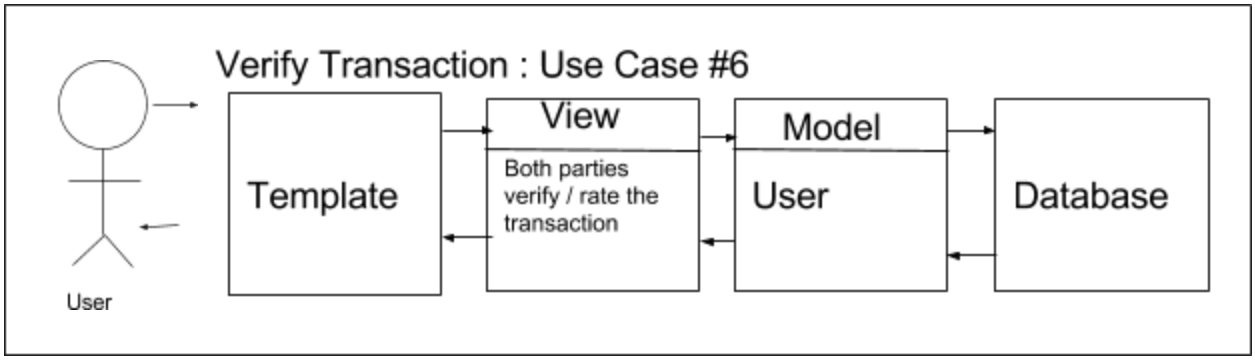







Select Payment Option : Use Case #4
User
Template
View
Select Payment Option
Model
User
Database


Redirected : Use Case #5
User
Template
View
User returns to home page if they cancel order
Model
User
Database


Verify Transaction : Use Case #6
User
Template
View
Both parties verify / rate the transaction
Model
User
Database

Appendix A – Agreement Between Customer and Contractor

The customer agrees to a textbook exchange system that includes searching, browsing and selling capabilities. See System Requirements Specification for more information. Additional features will be provided in further development intervals.

If there are future changes to this document, a new Customer and Contractor agreement will be drafted and signed by both parties.

Client

Name _____ Date _____
Print

Name _____ Date _____
Signature

Team

Name _____ Date _____
Print

Name _____ Date _____
Signature

Name _____ Date _____
Print

Name _____ Date _____
Signature

Name _____ Date _____
Signature

Appendix B – Team Review Sign-off

This document has been written by all members of the team. Additionally all team members have reviewed this document and agree on the content and format. Any disagreements or concerns are addressed in team comments below.

Team

Name _____ Date _____
Print
Name _____ Date _____
Signature
Comments
_____
_____
__

Name _____ Date _____
Print
Name _____ Date _____
Signature
Comments
_____
_____
__

Name _____ Date _____
Print
Name _____ Date _____
Signature
Comments
_____
_____
__

Name _____ Date _____
Print
Name _____ Date _____
Signature

Comments

_____

_____

__

Name _____ Date _____

Print

Name _____ Date _____

Signature

Comments

_____

_____

__

Appendix C – Document Contributions

Mehreen Awan drafted the requirements matrix, Daniel Kelly created the system architecture in part 2, John Gordon wrote the persistent data design of part 3, Mina Beshai wrote the introduction and Daniel Schomisch composed Appendix A and Appendix B.