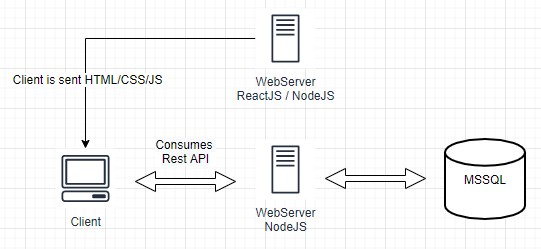
James Elam

SWDV 691-1

Service Layer Design

1. Technology

The Student Loan Calculator will consume a REST based API built on NodeJS with the ExpressJS framework. Users must first pass a username and password through /login in order to obtain a JSON web token to authenticate on subsequent requests. Most of the POST, PUT, and DELETE methods below will directly affect the analytics and charts on the page.



1. Error Response

The Service Layer will send appropriate HTTP Status Codes depending on the error that occurred. While the application will have client-side validation, alert messages will be triggered depending on the returned Status Code.

Forbidden - 403

Resource Not Found – 404

Invalid Input – 422

Too many requests – 429

1. Endpoints

a. POST /login

* 1. Post Body

{

userName: String password: String

}

* 1. Returns a JSON web Token that should be used by all requests for authentication.

{

accessToken: String

}

1. GET /me
   1. There are not many major use-cases for this endpoint but can be used for user-experience type message. ii. Returns basic information on the currently logged in user.

{

UserID: Number

UserName: String

Email: String

LastLogin: String

DateCreated: String

}

1. GET /me/student-loans
   1. Used for displaying a list of all of the user’s student loans and needed for a user to interact with them. ii. Returns an array of all user’s student loans currently not deleted

[{

LoanID: Number

LoanName: String

PaymentStart: String

LoanTerm: String

StartingPrinciple: Float

CurrentPrinciple: Float

AccruedInterest: Float

InterestRate: Float

MinimumPayment: Float

StatusID: Number

IsDeleted: Boolean

}, {…}, {…}]

d. POST /me/student-loans

i. Create a new student-loan. This will affect the analytics and charts on the page. ii. Post Body

{

LoanName: String

PaymentStart: String

LoanTerm: String

StartingPrinciple: Float

CurrentPrinciple: Float

AccruedInterest: Float

InterestRate: Float

MinimumPayment: Float

StatusID: Number

}

iii. The newly created student-loan is returned

{

LoanID: Number

LoanName: String

PaymentStart: String

LoanTerm: String

StartingPrinciple: Float

CurrentPrinciple: Float

AccruedInterest: Float

InterestRate: Float

MinimumPayment: Float

StatusID: Number

IsDeleted: Boolean

}

e. PUT /me/student-loan/:LoanID

1. Update an existing student loan.
2. The Post Body and returned entity is the same as the POST request.

1. DELETE /me/student-loans/:LoanID
   1. Deletes a single
   2. Returns a Boolean of whether or not the loan was deleted

{

LoanID: Number

Result: Boolean

}

1. GET /me/student-loans/payment-plans
   1. Returns all the payment-plans created by a user.

{

PaymentPlanID: Number

PlanName: String

IsCurrent: Boolean

IsDeleted: Boolean

Payments: [

{

PaymentID: Number

PaymentDate: String

PaymentAmount: Float

AllocationMethodID: Number

IsRecurring: Boolean

IsDeleted: Boolean

}, {…}, {…}

]

}

1. POST /me/student-loans/payment-plans
   * 1. Creates an empty payment-plan. ii.
     2. Post-Body

{

PlanName: Number

}

iii. Returns the same structure as GET

1. PUT /me/student-loans/payment-plans/:PaymentPlanID
   1. Updates an existing Payment Plans metadata
   2. Returns basic information on the currently logged in user.

1. DELETE /me/student-loans/payment-plans/:PaymentPlanID
   1. Deletes a payment plan and all payments associated with the plan.

{

PaymentPlanID: Number

Result: Boolean

}

1. POST /me/student-loans/payment-plans/:PaymentPlanID/Payment
   1. Creates a new payment for an existing payment plan
   2. Post-Data

{

PaymentID: Number

PaymentDate: String

PaymentAmount: Float

AllocationMethodID: Number

IsRecurring: Boolean

}

* 1. Returned data

{

PaymentID: Number

PaymentDate: String

PaymentAmount: Float

AllocationMethodID: Number

IsRecurring: Boolean

IsDeleted: Boolean

}

1. POST /me/student-loans/payment-plans/:PaymentPlanID/Payment/:PaymentID
   1. Deletes an existing payment associated with a PaymentPlanID
   2. Returned Data

{

PaymentPlanID: Number

Result: Boolean

}

1. GET /me/student-loans/aggregate-analytics
   1. Returns basic statistics calculated from a user’s student-loans. These are aggregate stats that change depending on the current plan selected

{

minimumPayOffDate: String estimatedPayoffDate: String currentPrincipalAmount: Float totalInterestPaidCurrent: Float totalInterestPaidExpected: Float averageInterestAccured: Float

}

1. GET /me/student-loans/chart-analytics
   1. Returns an array of data that will power the charts. Calculated from student-loans and current plan selected.

[{

Date: String

ExpectedTotalBalance: Float

Loans: [

{

LoanID: Number

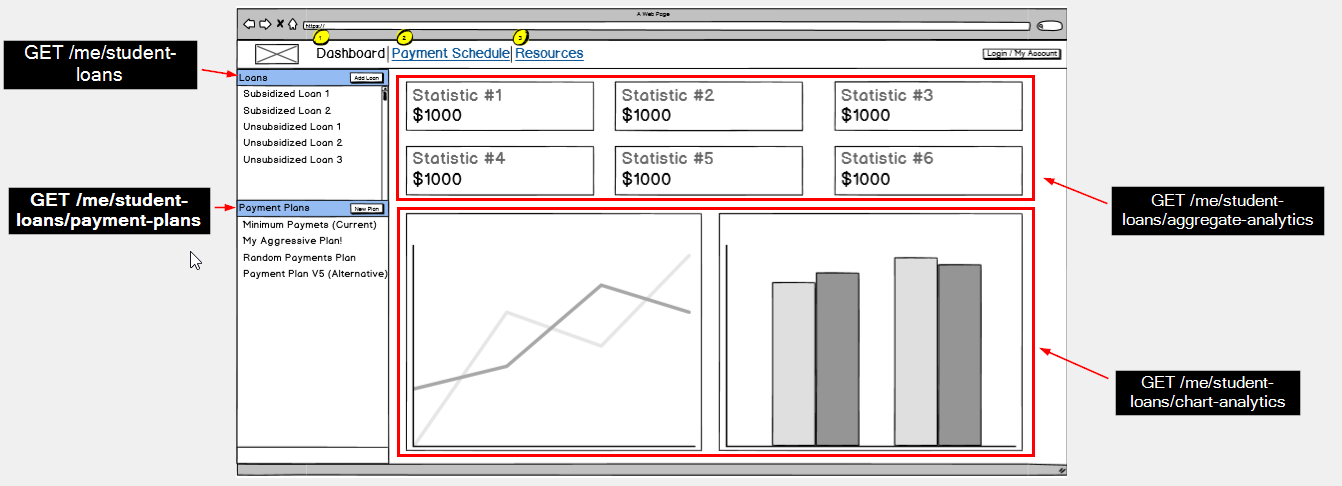
LoanName: String

ExpectedBalance: Number

}, {…}, {…}

]

}, {…}, {…}]

1. Endpoint Use Cases

