AVANTIKA UNIVERSITY, CSE

Technical Specification

Title	Avantika Store Management
Document Name	Technical specifications
Version No.	
Date	4-12-2021
Document Author(s)	Anannya Mital (AU19B1010)

Version History

Version	Date	Author	Change	Sections Affected
1.0				

Sponsor(s)

Reviewer(s)

Name	Title
Sir Ashish Bansal	Database Design
Sir Vrushal Verma	UI Design
Sir Kaliprasad Mahapatro	Application Working
Sir Sivaram Ramakrishnan	Application Working

Approver(s)

Name	Title
Sir Rushikesh Panwar	Overall Project
Sir Amarnath Solanki	Overall Project

Document References

Related Documents:
Functional specification document

Table of Contents

1. Introduction

This document will specify the technical specifications of the desktop application which is developed for Avantika Store Management. It will include all the technical aspects of the application like about the frameworks, test plan, validations, all the interaction in the application, etc.

Project Requirement

Desktop application for Avantika store management through which store manager can buy used items from customer, can give used items on loan, can get sales analysis, can filter and search records and can manage the store inventory through the application. The objectives of the application are:

1. To design an application which can decrease the time of manual entry in registers.
2. To Promote the use of reused products by new students.
3. To make some benefit to the students who bought the product and now are of no use.

1.1.1. Functional

- The system shall decrease the time taken for entry
- The system shall have minimum fields for entry.
- The system shall have less buffering time.
- The system shall store the data on submit in the database.

- The fields shall not be time taking.
- The system shall send some type of email to promote reusing of products
- The system shall sell and buy the reused products
- The system shall give the product on loan to other users for a specific time.
- The system shall notify store manager if the time limit exceeds.
- The system shall generate a unique id for every reusable product.

1.1.2. Non-Functional

- The system shall filter the tables according to the need.
- The filter combination should work properly and fast.
- The form field should be less time taking.
- The application should work fast.
- There should be a feature of going from one page to every page.
- There should be only one admin i.e., store manager.
- It should store huge amount of data for transaction history.
- It should be user-friendly.

2. Technical Design

In this topic we will discuss about frameworks and database design. For framework we have two options electron and QT from which we have to choose one. Another thing was we have to decide that which database should be take for the desktop application. Again, the options were only two, MySQL and SQLite.

Technical Approach

For the framework we had chosen electron because we were used to HTML, CSS and JavaScript because we had developed some web applications in earlier academic projects. Application functionality will be easy foe us in JavaScript and the CSS also.

Scope

2.1.1. In Scope

We can describe the project scopes as the part of project planning that involves determining and documenting a list of specific project goals, deliverables, tasks, costs and deadlines. The project scope for this desktop application will be:

- Store manager can buy used items from customers (Faculty and students).
- Store manager can give used items on loan.
- Store manager can get regular sales analysis.
- Store manager can filter and search records.
- Store manager can manage the store inventory through application.

2.1.2. Out of Scope

- Notifications can be sent to defaulters.
- Connectivity to internet can be done.
- Add, delete and edit of store products through the application.
- Cash payment can be avoided.

Assumptions

- Assumed that they have a fast processor for better performance of the applications.
- Assumed that xampp is installed for database connectivity.
- They know the login password every time.

Issues

- Notifications can be sent to defaulters.
- Connectivity to internet can be done.
- Add, delete and edit of store products through the application.
- Cash payment can be avoided.

Risks

- If the admin forgets the login password then he has to go to the higher authorities for this issue.
- If the application crashes then admin has to again install it.
- Some issues can arise regarding database connectivity because maybe they don’t have xampp installed on their desktops.

2.1.3. Code

Sl.	Object	Event	Logic
01	setDate()	Show date and time	Fetching date and time from device and displaying it in an input field.
02	Login()	Check password validation	It will fetch the admin password from database and compare it with user entry.
03	Connect()	Connects the database.	We give the details of the database and it connects with it.
04	Validid()	Will check the customer id is valid or not	If the customer is student the id should be of 9 digits and for faculty it should be of 5 digits.
05	Insert_student()	It inserts the details in database	Whenever a student will come to buy any product, after it entering it will send the details to the database.
06	Insert_faculty()	It inserts the details in database	Whenever a faculty will come to buy any product, after it entering it will send the details to the database.
07	Autoname()	It will the name automatically	After entering the enrollment id it will automatically fill the customer name in the mane field.
08	Date_filer()	Filter by date	It will filter the rows according to the date entered.
09	myFunction()	Filter by ID	It will filter the table data by the customer ID admin enters.
10	Returnpro()	It will return the product	This button will return the product in the database with the product and loan id.

2.1.4. Field

Field Name	Short Name	Long Name	Data Type	Format	Len	Translate Y/N
Password	Password	Password	Password	Input	10	N
Username	Username	Username	Text	Text	5	Y
Customer	Customer	Customer	Text	Dropdown		Y
Enrollment ID	ID	Enrollment ID	Alpha-numeric	Input	9	Y
Name	Name	Customer Name	Text	Input	25	Y
Product Type	Type	Product Type	Text	Dropdown	3	Y
Product ID	Product ID	Product ID	Alpha-numeric	Select	30	Y
Quantity	QTY	Quantity	Number	Input	NA	Y
Cost	Cost	Cost	Number	Input	NA	Y
Date and time	DT	Date Time	Date time	Fixed	NA	Y
Duration	Duration	Duration	Number	Input	NA	Y
Selling Cost	Selling Cost	Selling Cost	Number	Input	NA	Y

2.1.5. Message Catalog

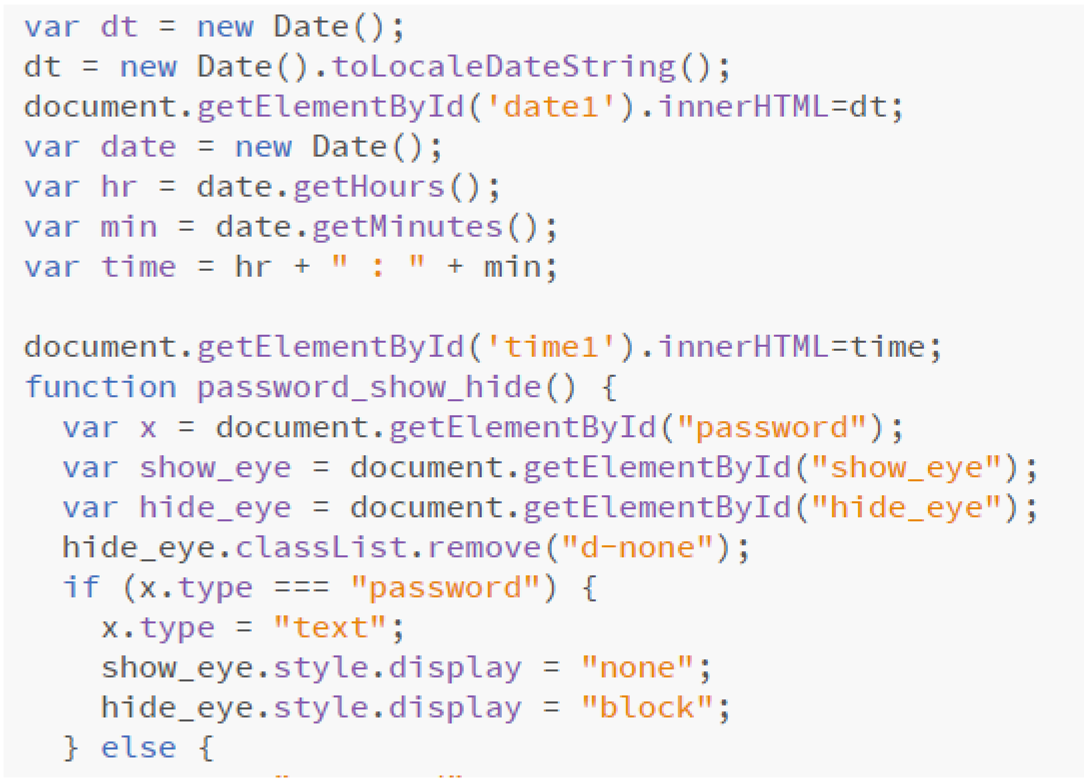
Set Nbr	Msg Nbr	Severity	Description	Explain Text
01	01	Medium	It will confirm the form submission	Do you want to confirm the form details and want to continue?
02	02	Medium	It will come when the admin return the product.	Product returned!

2.1.6. Security Requirements

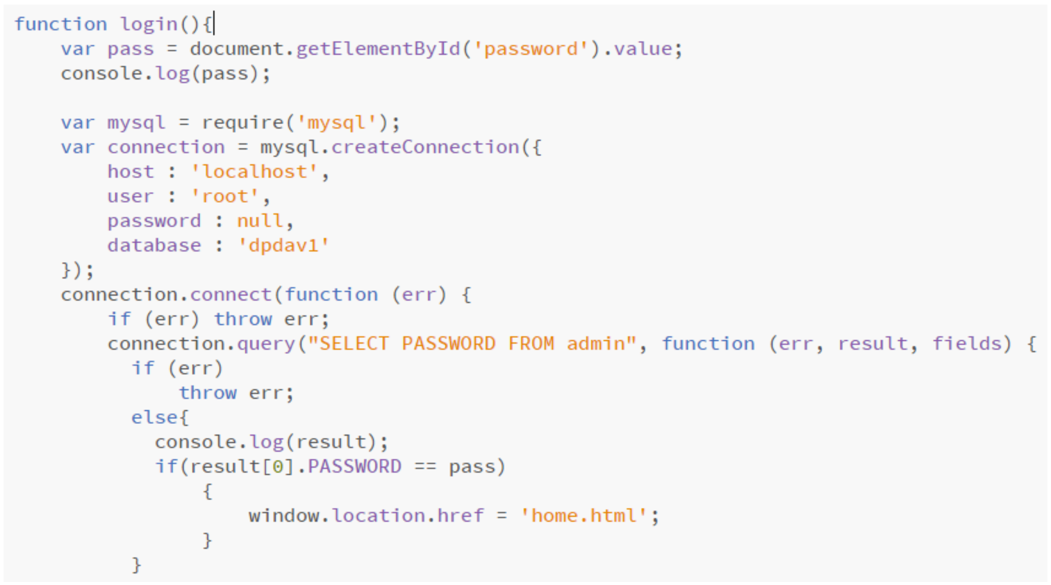
Object Name	Navigation / Portal Tab	Roles / Permission List	Access Type
Login	Dashboard	Store Manager	Admin

2.1.6.1. Programming Logic

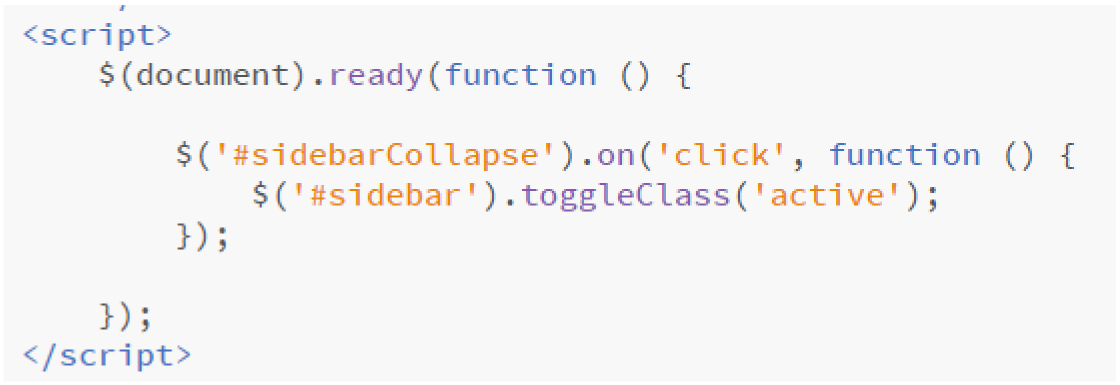
The programming logic is used for JavaScript for fetching the data from database and for sending the data in the database.



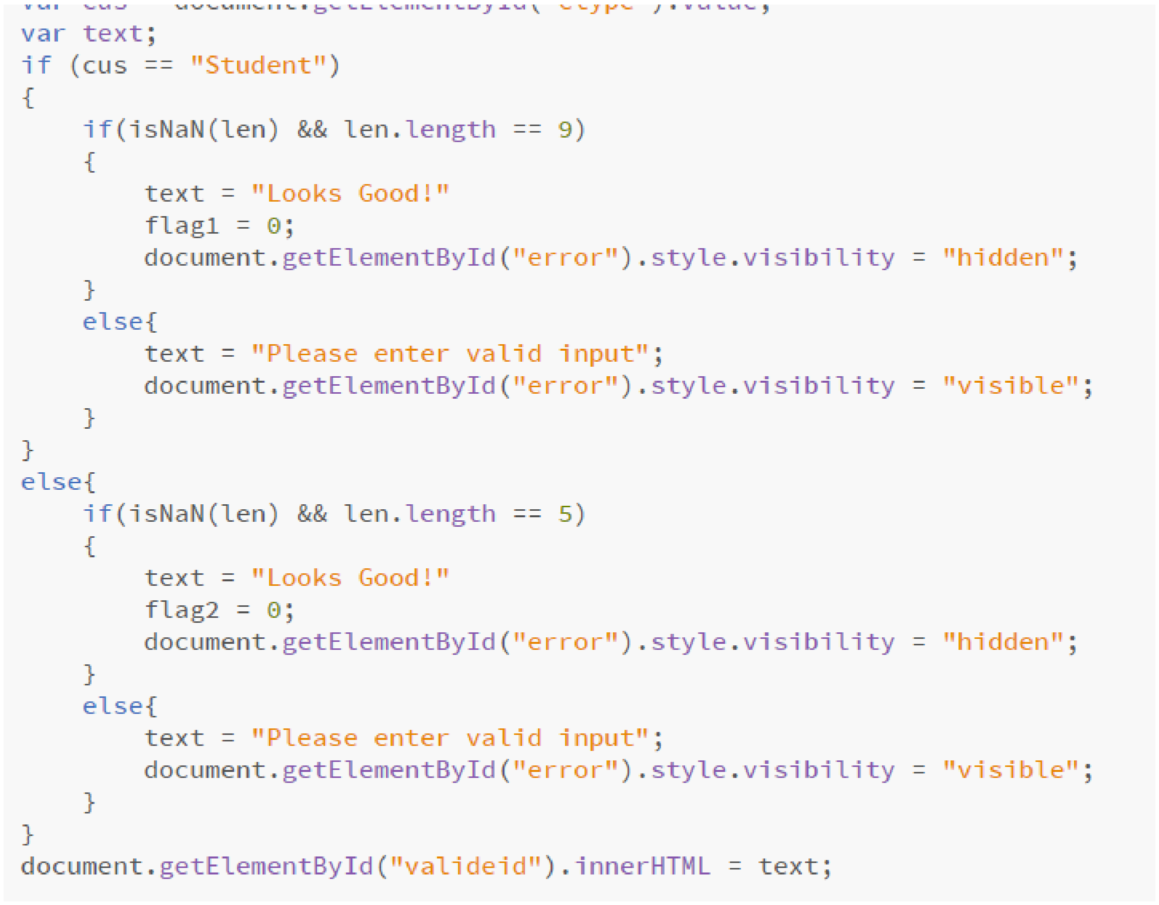
```
var dt = new Date();
dt = new Date().toLocaleDateString();
document.getElementById('date1').innerHTML=dt;
var date = new Date();
var hr = date.getHours();
var min = date.getMinutes();
var time = hr + " : " + min;

document.getElementById('time1').innerHTML=time;
function password_show_hide() {
  var x = document.getElementById("password");
  var show_eye = document.getElementById("show_eye");
  var hide_eye = document.getElementById("hide_eye");
  hide_eye.classList.remove("d-none");
  if (x.type === "password") {
    x.type = "text";
    show_eye.style.display = "none";
    hide_eye.style.display = "block";
  } else {
```



```
function login(){
    var pass = document.getElementById('password').value;
    console.log(pass);

    var mysql = require('mysql');
    var connection = mysql.createConnection({
        host : 'localhost',
        user : 'root',
        password : null,
        database : 'dpdav1'
    });
    connection.connect(function (err) {
        if (err) throw err;
        connection.query("SELECT PASSWORD FROM admin", function (err, result, fields) {
          if (err)
              throw err;
          else{
            console.log(result);
            if(result[0].PASSWORD == pass)
                {
                    window.location.href = 'home.html';
                }
          }
```



```
<script>
    $(document).ready(function () {

        $('#sidebarCollapse').on('click', function () {
            $('#sidebar').toggleClass('active');
        });

    });
</script>
```



```
var cus = document.getElementById("ctype").value;
var text;
if (cus == "Student")
{
    if(isNaN(len) && len.length == 9)
    {
        text = "Looks Good!"
        flag1 = 0;
        document.getElementById("error").style.visibility = "hidden";
    }
    else{
        text = "Please enter valid input";
        document.getElementById("error").style.visibility = "visible";
    }
}
else{
    if(isNaN(len) && len.length == 5)
    {
        text = "Looks Good!"
        flag2 = 0;
        document.getElementById("error").style.visibility = "hidden";
    }
    else{
        text = "Please enter valid input";
        document.getElementById("error").style.visibility = "visible";
    }
}
document.getElementById("valideid").innerHTML = text;
```

```
function insert_student(){
    var eid = document.getElementById('inputid').value;
    var cname = document.getElementById('inputcname').value;
    var pid = document.getElementById('inputpid').value;
    var qty = document.getElementById('inputqty').value;
    var cost = document.getElementById('inputcost').value;
    var dt = document.getElementById('inputdt').value;
    connection.query("INSERT INTO `transactions`(`S_ID`, `Name`, `P_ID`, `QTY`, `AMOUNT`, `DT`,
    `T_TYPE`, `A_USERNAME`) VALUES
    ('"+eid+"','"+cname+"',"+pid+","+qty+","+cost+",'"+dt+"','sell','Amarnath')", function
    (err, result, fields) {
      if (err)
          throw err;
      else{
        console.log("Success");
      }
    });
    connection.query("update products p SET p.QTY = p.QTY - '"+qty+"' where p.P_ID =
    '"+pid+"';", function (err, result, fields) {
      if (err)
          throw err;
      else{
        console.log("Successqty");
      }
    });
```

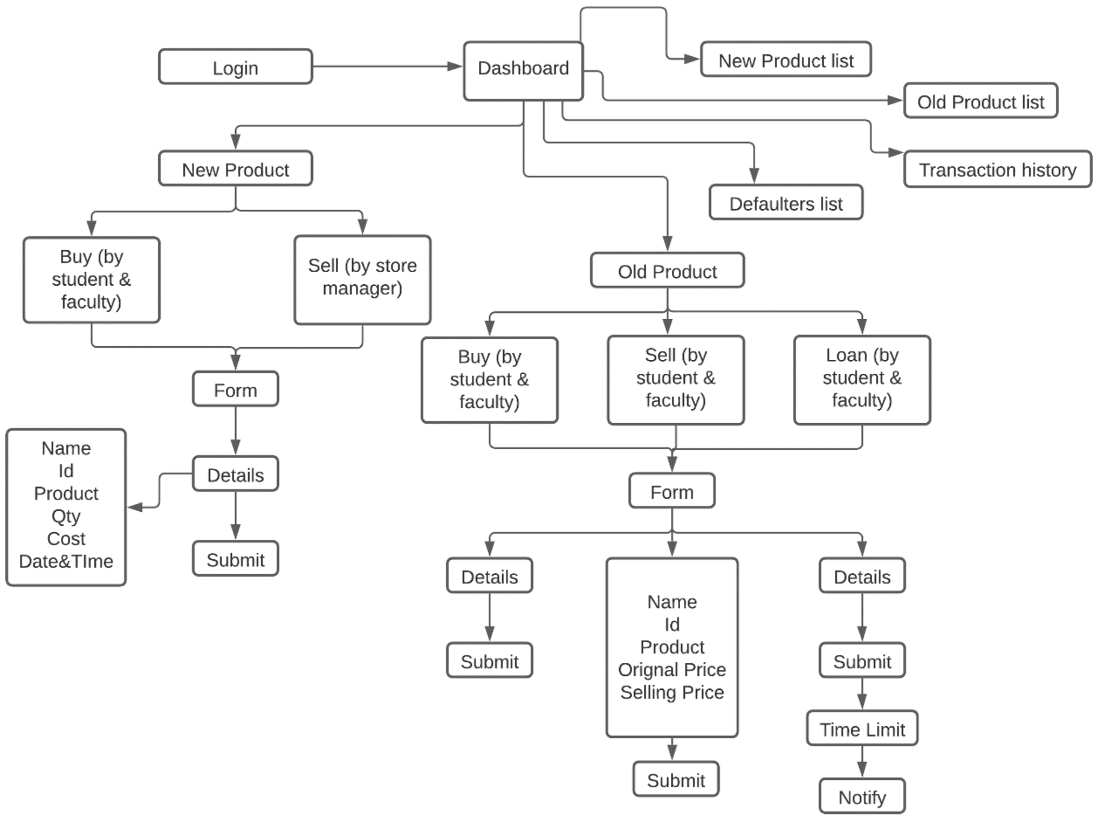
```
function autoname(){
        var eid = document.getElementById('inputid').value;
        connection.query("Select * from students where S_ID = '"+eid+"';", function (err, result,
        fields) {
          if (err)
              throw err;
          else{
            console.log(result);
            var string = JSON.stringify(result);
            var json = JSON.parse(string);
            var name = json[0].S_NAME;
            document.getElementById('inputcname').value = name;
          }
        });
    }
```

```
function insert_new(){
    var customer = document.getElementById('ctype').value;
    if(customer == "Student")
        {
            insert_student();
        }
    else{
        insert_faculty();
    }
}
</script>
```

```
function date_filer(){
    var date = document.getElementById('namesearch').value;
    console.log("date =", date);
    table = document.getElementById("return-table");
    tr = table.getElementsByTagName("tr");
    for (i = 0; i < tr.length; i++) {
      td = tr[i].getElementsByTagName("td")[1];
      if (td) {
        txtValue = td.textContent || td.innerText;
          console.log("textValue",txtValue);
        if (txtValue.indexOf(date) > -1) {
          tr[i].style.display = "";
        } else {
            console.log("Hellooooo");
          tr[i].style.display = "none";
        }
      }
    }
```

```
function myFunction() {
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("idsearch");
    filter = input.value.toUpperCase();
    table = document.getElementById("return-table");
    tr = table.getElementsByTagName("tr");
    for (i = 0; i < tr.length; i++) {
      td = tr[i].getElementsByTagName("td")[0];
      if (td) {
        txtValue = td.textContent || td.innerText;
          console.log("textValue",txtValue);
        if (txtValue.toUpperCase().indexOf(filter) > -1) {
          tr[i].style.display = "";
        } else {
            console.log("Hellooooo");
          tr[i].style.display = "none";
        }
      }
    }
```

2.1.6.2. Process Flow Diagram



3. Testing

After implementation we have to perform testing of the desktop application. We have to check all the validation from login to filling the form and also the confirmations.

Test Plan

Test ID	Test Title
01	Login with correct password
02	Error with wrong password.
03	Error on submitting null fields.
04	Error on filling wrong employee ID
05	Error on filling wrong enrollment ID
06	Form submission on entering correct data
07	Delete the loan data after clicking on return button
08	Enrollment ID filter Should work
09	Product ID Filter should work

Test Data

Test ID	Test Data	Status
01	amarnath	Pass
02	Vsdvs	Fail
03	Null data	Pass
04	AU12	Pass
05	AU19B101	Pass
06	Entered all data correctly	Pass
07	Delete the loan data after clicking on return button	Pass
08	AU19B1010	Pass
09	3	Pass

Third Party Requirements

- Notifications can be sent to defaulters.
- Add, delete and edit of store products through the application.
- Cash payment can be avoided.
- Other Management members can also access the system.

4. Issues

Issue	**Status**
Wrong password error	Unresolved
Student Registration	Resolved
Sending Notifications	Unresolved