LoanReport.cshtml

```
@model List<BankLoanProject.Models.LoanReportViewModel>
@{
  var role = Context.Session.GetString("Role");
  Layout = null;
}

<!-- Loan Report Section -->
<div id="loan" class="report-section table-container">
  <div class="section-header">
    <div class="section-info">
      <span class="badge bg-primary mb-2">Loan Reports</span>
      <h4 class="mb-2">
        <i class="fas fa-file-invoice-dollar me-2"></i>
        Loan Report
      </h4>
      <p class="text-muted">Comprehensive loan application reports and analytics.</p>
    </div>
    <div class="export-filter-controls">
      <button class="btn btn-export" id="btnExport">
        <i class="fas fa-file-pdf"></i> Export PDF
      </button>
      <!-- Filter Dropdown -->
      <div class="dropdown">
        <button class="btn btn-filter dropdown-toggle" type="button" data-bs-toggle="dropdown">
          <i class="fas fa-filter"></i> Filter
```

```
            </button>
            <div class="dropdown-menu p-3">
              <label class="fw-bold">By Loan Type:</label>
              <select id="loanTypeFilter" class="form-control mb-2">
                <option value="">All</option>
                <option value="Home">Home</option>
                <option value="Vehicle">Vehicle</option>
                <option value="Personal">Personal</option>
              </select>

              <label class="fw-bold">By Application Date:</label>
              <input type="date" id="applicationDateFilter" class="form-control mb-2">

              <label class="fw-bold">By Status:</label>
              <select id="statusFilter" class="form-control mb-2">
                <option value="">All</option>
                <option value="Approved">Approved</option>
                <option value="Pending">Pending</option>
              </select>

              <button class="btn btn-success mt-2 w-100"
id="btnApplyFilter">Apply</button>
              <button class="btn btn-danger mt-2 w-100"
id="btnClearFilter">Clear</button>
            </div>
          </div>
        </div>
      </div>
```

```
  <!-- Table Controls - Fixed Layout -->
  <div class="table-controls">
    <!-- Left side - Rows per page -->
    <div class="rows-control">
      <label for="rowsPerPage">
        <i class="fas fa-list-ol"></i>
        Rows per page:
      </label>
      <select id="rowsPerPage" class="form-select"
onchange="changeRowsPerPage()">
        <option value="5" selected>5</option>
        <option value="10">10</option>
        <option value="15">15</option>
        <option value="20">20</option>
      </select>
    </div>

    <!-- Right side - Search -->
    <div class="search-control">
      <div class="input-group search-input-group">
        <span class="input-group-text">
          <i class="fas fa-search"></i>
        </span>
        <input type="text" class="form-control" id="searchInput" placeholder="Search
loan reports..." oninput="searchReports()">
      </div>
    </div>
  </div>
```

```
<div class="table-wrapper">
  <table class="table" id="loanReportTable">
    <thead>
      <tr>
        @if (role == "Admin")
        {
          <th><i class="fas fa-user"></i> Customer ID</th>
          <th><i class="fas fa-user-tag"></i> Customer Name</th>
        }
        <th><i class="fas fa-tag"></i> Product Type</th>
        <th><i class="fas fa-file-signature"></i> Product Name</th>
        <th><i class="fas fa-money-bill-wave"></i> Loan Amount</th>
        <th><i class="fas fa-calendar-alt"></i> Application Date</th>
        <th><i class="fas fa-check-circle"></i> Status</th>
      </tr>
    </thead>
    <tbody class="report-table-body">
      @foreach (var report in Model)
      {
        <tr class="report-row">
          @if (role == "Admin")
          {
            <td>@report.CustomerId</td>
            <td><strong>@report.CustomerName</strong></td>
          }
          <td>@report.ProductType</td>
          <td>@report.ProductName</td>
```

```
                <td><span
class="amount">@report.LoanAmount.ToString("C")</span></td>
                <td>@report.ApplicationDate.ToString("dd-MM-yyyy")</td>
                <td><span class="@(report.ApprovalStatus == "Approved" ? "status-
approved" : "status-pending")">@report.ApprovalStatus</span></td>
            </tr>
          }
        </tbody>
      </table>
    </div>

    <!-- Pagination Section -->
    <div class="pagination-section">
      <div class="pagination-info" id="loanPaginationInfo">
        <!-- Pagination info will be generated by JavaScript -->
      </div>
      <nav aria-label="Loan report pagination">
        <ul class="pagination" id="loanPaginationContainer">
          <!-- Pagination buttons will be generated by JavaScript -->
        </ul>
      </nav>
    </div>
</div>

<script>
    $(document).ready(function () {
      // Initialize pagination
      setTimeout(initializePagination, 100);
```

```
    // Export Table as PDF
    $("#btnExport").click(function () {
        exportToPDF('loanReportTable', 'Loan_Report');
    });

    // Apply Filters
    $("#btnApplyFilter").click(function () {
        applyFilters();
    });

    // Clear Filters
    $("#btnClearFilter").click(function () {
        clearFilters();
    });
});

// Global variables for pagination
let currentPage = 1;
let rowsPerPage = 5;
let allRows = [];
let filteredRows = [];

function initializePagination() {
    allRows = Array.from(document.querySelectorAll('.report-row'));
    filteredRows = [...allRows];
    updateTable();
}
```

```
function applyFilters() {
    let selectedLoanType = $("#loanTypeFilter").val().toLowerCase();
    let selectedStatus = $("#statusFilter").val().toLowerCase();
    let applicationDate = $("#applicationDateFilter").val();

    filteredRows = allRows.filter(row => {
        const cells = row.querySelectorAll('td');
        const role = '@role';
        let loanType, status, rowApplicationDate;

        if (role === "Admin") {
            loanType = cells[2].textContent.trim().toLowerCase();
            status = cells[6].textContent.trim().toLowerCase();
            rowApplicationDate = cells[5].textContent.trim();
        } else {
            loanType = cells[0].textContent.trim().toLowerCase();
            status = cells[4].textContent.trim().toLowerCase();
            rowApplicationDate = cells[3].textContent.trim();
        }

        let showRow = true;

        if (selectedLoanType && loanType !== selectedLoanType) {
            showRow = false;
        }
        if (selectedStatus && status !== selectedStatus) {
            showRow = false;
        }
```

```
        if (applicationDate) {
            let dateParts = rowApplicationDate.split("-");
            let formattedRowDate = dateParts[2] + "-" + dateParts[1] + "-" + dateParts[0];
            if (applicationDate !== formattedRowDate) {
                showRow = false;
            }
        }

        return showRow;
    });

    currentPage = 1;
    updateTable();
}

function clearFilters() {
    $("#loanTypeFilter").val("");
    $("#statusFilter").val("");
    $("#applicationDateFilter").val("");
    $("#searchInput").val("");

    filteredRows = [...allRows];
    currentPage = 1;
    updateTable();
}

function searchReports() {
    const searchTerm = document.getElementById('searchInput').value.toLowerCase();
```

```
    if (searchTerm === '') {
        filteredRows = [...allRows];
    } else {
        filteredRows = allRows.filter(row => {
            const text = row.textContent.toLowerCase();
            return text.includes(searchTerm);
        });
    }

    currentPage = 1;
    updateTable();
}

function changeRowsPerPage() {
    rowsPerPage = parseInt(document.getElementById('rowsPerPage').value);
    currentPage = 1;
    updateTable();
}

function updateTable() {
    // Hide all rows first
    allRows.forEach(row => row.style.display = 'none');

    // Calculate pagination
    const totalRows = filteredRows.length;
    const totalPages = Math.ceil(totalRows / rowsPerPage) || 1;
    const startIndex = (currentPage - 1) * rowsPerPage;
```

```
    const endIndex = Math.min(startIndex + rowsPerPage, totalRows);

    // Show current page rows
    for (let i = startIndex; i < endIndex; i++) {
        if (filteredRows[i]) {
            filteredRows[i].style.display = '';
        }
    }

    // Update pagination controls
    updatePagination(totalPages, totalRows, startIndex, endIndex);
}

function updatePagination(totalPages, totalRows, startIndex, endIndex) {
    const paginationContainer = document.getElementById('loanPaginationContainer');
    const paginationInfo = document.getElementById('loanPaginationInfo');

    // Clear existing pagination
    paginationContainer.innerHTML = '';

    if (totalPages <= 1) {
        const singleBtn = createPaginationButton('1', 1, false, true);
        paginationContainer.appendChild(singleBtn);
    } else {
        // First page button
        const firstBtn = createPaginationButton('«', 1, currentPage === 1);
        paginationContainer.appendChild(firstBtn);
```

```
        // Previous page button
        const prevBtn = createPaginationButton('‹', currentPage - 1, currentPage === 1);
        paginationContainer.appendChild(prevBtn);

        // Page number buttons
        const maxVisiblePages = 5;
        let startPage = Math.max(1, currentPage - Math.floor(maxVisiblePages / 2));
        let endPage = Math.min(totalPages, startPage + maxVisiblePages - 1);

        if (endPage - startPage + 1 < maxVisiblePages) {
          startPage = Math.max(1, endPage - maxVisiblePages + 1);
        }

        for (let i = startPage; i <= endPage; i++) {
          const pageBtn = createPaginationButton(i, i, false, i === currentPage);
          paginationContainer.appendChild(pageBtn);
        }

        // Next page button
        const nextBtn = createPaginationButton('›', currentPage + 1, currentPage ===
totalPages);
        paginationContainer.appendChild(nextBtn);

        // Last page button
        const lastBtn = createPaginationButton('»', totalPages, currentPage ===
totalPages);
        paginationContainer.appendChild(lastBtn);
    }
```

```
    // Update pagination info
    if (totalRows > 0) {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            Showing ${startIndex + 1} to ${endIndex} of ${totalRows} entries
        `;
    } else {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            No matching records found
        `;
    }
}

function createPaginationButton(text, page, disabled, active = false) {
    const li = document.createElement('li');
    li.className = ` page-item ${active ? 'active' : ''} ${disabled ? 'disabled' : ''}` ;

    const button = document.createElement('a');
    button.className = 'page-link';
    button.innerHTML = text;
    button.href = '#';

    if (!disabled) {
        button.onclick = (e) => {
            e.preventDefault();
            goToPage(page);
        };
```

```
    }

    li.appendChild(button);
    return li;
}

function goToPage(page) {
    currentPage = page;
    updateTable();
}

function exportToPDF(tableId, filename) {
    const { jsPDF } = window.jspdf;
    const doc = new jsPDF();

    // Add title
    doc.setFontSize(16);
    doc.text('Loan Report', 14, 15);

    // Add date
    doc.setFontSize(10);
    doc.text('Generated on: ' + new Date().toLocaleDateString(), 14, 25);

    // Get table data
    const table = document.getElementById(tableId);
    const headers = [];
    const data = [];
```

```
// Extract headers
table.querySelectorAll('thead th').forEach(th => {
  headers.push(th.textContent.trim());
});

// Extract visible rows data
table.querySelectorAll('tbody tr').forEach(tr => {
  if (tr.style.display !== 'none') {
    const row = [];
    tr.querySelectorAll('td').forEach(td => {
      row.push(td.textContent.trim());
    });
    data.push(row);
  }
});

// Generate table
doc.autoTable({
  head: [headers],
  body: data,
  startY: 35,
  styles: {
    fontSize: 8,
    cellPadding: 2
  },
  headStyles: {
    fillColor: [44, 62, 80],
    textColor: 255
```

```
        }
      });

      // Save the PDF
      const timestamp = new Date().toISOString().replace(/[-:.]/g, "");
      doc.save(`${filename}_${timestamp}.pdf`);
    }
</script>
```

OutstandingReport.cshtml

```
@model List<BankLoanProject.Models.OutstandingReportViewModel>
@{
    var role = Context.Session.GetString("Role");
    Layout = null;
}

<!-- Outstanding Report Section -->
<div id="outstanding" class="report-section table-container">
    <div class="section-header">
        <div class="section-info">
            <span class="badge bg-warning mb-2">Outstanding Reports</span>
            <h4 class="mb-2">
                <i class="fas fa-money-bill-wave me-2"></i>
                Outstanding Loans Report
            </h4>
            <p class="text-muted">Track outstanding loan balances and payment status.</p>
        </div>
        <div class="export-filter-controls">
```

```
        <button class="btn btn-export" id="btnExport">
          <i class="fas fa-file-pdf"></i> Export PDF
        </button>
        <!-- Filter Dropdown -->
        <div class="dropdown">
          <button class="btn btn-filter dropdown-toggle" type="button" data-bs-
toggle="dropdown">
            <i class="fas fa-filter"></i> Filter
          </button>
          <div class="dropdown-menu p-3">
            <label class="fw-bold">By Loan Type:</label>
            <select id="loanTypeFilter" class="form-control mb-2">
              <option value="">All</option>
              <option value="Home">Home</option>
              <option value="Vehicle">Vehicle</option>
              <option value="Personal">Personal</option>
            </select>

            <button class="btn btn-success mt-2 w-100"
id="btnApplyFilter">Apply</button>
            <button class="btn btn-danger mt-2 w-100"
id="btnClearFilter">Clear</button>
          </div>
        </div>
      </div>
    </div>

    <!-- Table Controls - Fixed Layout -->
    <div class="table-controls">
```

```
    <!-- Left side - Rows per page -->
    <div class="rows-control">
      <label for="rowsPerPage">
        <i class="fas fa-list-ol"></i>
        Rows per page:
      </label>
      <select id="rowsPerPage" class="form-select"
onchange="changeRowsPerPage()">
        <option value="5" selected>5</option>
        <option value="10">10</option>
        <option value="15">15</option>
        <option value="20">20</option>
      </select>
    </div>

    <!-- Right side - Search -->
    <div class="search-control">
      <div class="input-group search-input-group">
        <span class="input-group-text">
          <i class="fas fa-search"></i>
        </span>
        <input type="text" class="form-control" id="searchInput" placeholder="Search
outstanding reports..." oninput="searchReports()">
      </div>
    </div>
  </div>

  <div class="table-wrapper">
    <table class="table" id="outstandingReportTable">
```

```
        <thead>
          <tr>
            <th><i class="fas fa-hashtag"></i> Loan ID</th>
            @if (role == "Admin")
            {
              <th><i class="fas fa-user"></i> Customer ID</th>
              <th><i class="fas fa-user-tag"></i> Customer Name</th>
            }
            <th><i class="fas fa-tag"></i> Product Type</th>
            <th><i class="fas fa-file-signature"></i> Product Name</th>
            <th><i class="fas fa-money-bill-wave"></i> Loan Amount</th>
            <th><i class="fas fa-exclamation-triangle"></i> Outstanding Balance</th>
          </tr>
        </thead>
        <tbody class="report-table-body">
          @foreach (var report in Model)
          {
            <tr class="report-row">
              <td><strong>@report.LoanId</strong></td>
              @if (role == "Admin")
              {
                <td>@report.CustomerId</td>
                <td><strong>@report.CustomerName</strong></td>
              }
              <td>@report.ProductType</td>
              <td>@report.ProductName</td>
              <td><span
class="amount">@report.LoanAmount.ToString("C")</span></td>
```

```
                <td><span
class="amount">@report.OutstandingBalance.ToString("C")</span></td>
            </tr>
          }
        </tbody>
      </table>
    </div>

    <!-- Pagination Section -->
    <div class="pagination-section">
      <div class="pagination-info" id="outstandingPaginationInfo">
        <!-- Pagination info will be generated by JavaScript -->
      </div>
      <nav aria-label="Outstanding report pagination">
        <ul class="pagination" id="outstandingPaginationContainer">
          <!-- Pagination buttons will be generated by JavaScript -->
        </ul>
      </nav>
    </div>
</div>

<script>
    $(document).ready(function () {
      // Initialize pagination
      setTimeout(initializePagination, 100);

      // Export Table as PDF
      $("#btnExport").click(function () {
```

```
        exportToPDF('outstandingReportTable', 'Outstanding_Loans_Report');
    });

    // Apply Filters
    $("#btnApplyFilter").click(function () {
        applyFilters();
    });

    // Clear Filters
    $("#btnClearFilter").click(function () {
        clearFilters();
    });
});

// Global variables for pagination
let currentPage = 1;
let rowsPerPage = 5;
let allRows = [];
let filteredRows = [];

function initializePagination() {
    allRows = Array.from(document.querySelectorAll('.report-row'));
    filteredRows = [...allRows];
    updateTable();
}

function applyFilters() {
    let selectedLoanType = $("#loanTypeFilter").val().toLowerCase();
```

```
    filteredRows = allRows.filter(row => {
        const cells = row.querySelectorAll('td');
        const role = '@role';
        let loanType;

        if (role === "Admin") {
            loanType = cells[3].textContent.trim().toLowerCase();
        } else {
            loanType = cells[1].textContent.trim().toLowerCase();
        }

        let showRow = true;

        if (selectedLoanType && loanType !== selectedLoanType) {
            showRow = false;
        }

        return showRow;
    });

    currentPage = 1;
    updateTable();
}

function clearFilters() {
    $("#loanTypeFilter").val("");
    $("#searchInput").val("");
```

```
    filteredRows = [...allRows];
    currentPage = 1;
    updateTable();
}

function searchReports() {
    const searchTerm = document.getElementById('searchInput').value.toLowerCase();

    if (searchTerm === '') {
        filteredRows = [...allRows];
    } else {
        filteredRows = allRows.filter(row => {
            const text = row.textContent.toLowerCase();
            return text.includes(searchTerm);
        });
    }

    currentPage = 1;
    updateTable();
}

function changeRowsPerPage() {
    rowsPerPage = parseInt(document.getElementById('rowsPerPage').value);
    currentPage = 1;
    updateTable();
}
```

```
  function updateTable() {
    // Hide all rows first
    allRows.forEach(row => row.style.display = 'none');

    // Calculate pagination
    const totalRows = filteredRows.length;
    const totalPages = Math.ceil(totalRows / rowsPerPage) || 1;
    const startIndex = (currentPage - 1) * rowsPerPage;
    const endIndex = Math.min(startIndex + rowsPerPage, totalRows);

    // Show current page rows
    for (let i = startIndex; i < endIndex; i++) {
      if (filteredRows[i]) {
        filteredRows[i].style.display = '';
      }
    }

    // Update pagination controls
    updatePagination(totalPages, totalRows, startIndex, endIndex);
  }

  function updatePagination(totalPages, totalRows, startIndex, endIndex) {
    const paginationContainer =
document.getElementById('outstandingPaginationContainer');
    const paginationInfo = document.getElementById('outstandingPaginationInfo');

    // Clear existing pagination
    paginationContainer.innerHTML = '';
```

```
if (totalPages <= 1) {
    const singleBtn = createPaginationButton('1', 1, false, true);
    paginationContainer.appendChild(singleBtn);
} else {
    // First page button
    const firstBtn = createPaginationButton('«', 1, currentPage === 1);
    paginationContainer.appendChild(firstBtn);

    // Previous page button
    const prevBtn = createPaginationButton('‹', currentPage - 1, currentPage === 1);
    paginationContainer.appendChild(prevBtn);

    // Page number buttons
    const maxVisiblePages = 5;
    let startPage = Math.max(1, currentPage - Math.floor(maxVisiblePages / 2));
    let endPage = Math.min(totalPages, startPage + maxVisiblePages - 1);

    if (endPage - startPage + 1 < maxVisiblePages) {
        startPage = Math.max(1, endPage - maxVisiblePages + 1);
    }

    for (let i = startPage; i <= endPage; i++) {
        const pageBtn = createPaginationButton(i, i, false, i === currentPage);
        paginationContainer.appendChild(pageBtn);
    }

    // Next page button
```

```
        const nextBtn = createPaginationButton('›', currentPage + 1, currentPage === totalPages);
        paginationContainer.appendChild(nextBtn);

        // Last page button
        const lastBtn = createPaginationButton('»', totalPages, currentPage === totalPages);
        paginationContainer.appendChild(lastBtn);
    }

    // Update pagination info
    if (totalRows > 0) {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            Showing ${startIndex + 1} to ${endIndex} of ${totalRows} entries
        `;
    } else {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            No matching records found
        `;
    }
}

function createPaginationButton(text, page, disabled, active = false) {
    const li = document.createElement('li');
    li.className = `page-item ${active ? 'active' : ''} ${disabled ? 'disabled' : ''}`;

    const button = document.createElement('a');
```

```
    button.className = 'page-link';
    button.innerHTML = text;
    button.href = '#';

    if (!disabled) {
      button.onclick = (e) => {
        e.preventDefault();
        goToPage(page);
      };
    }

    li.appendChild(button);
    return li;
}

function goToPage(page) {
    currentPage = page;
    updateTable();
}

function exportToPDF(tableId, filename) {
    const { jsPDF } = window.jspdf;
    const doc = new jsPDF();

    // Add title
    doc.setFontSize(16);
    doc.text('Outstanding Loans Report', 14, 15);
```

```
// Add date
doc.setFontSize(10);
doc.text('Generated on: ' + new Date().toLocaleDateString(), 14, 25);

// Get table data
const table = document.getElementById(tableId);
const headers = [];
const data = [];

// Extract headers
table.querySelectorAll('thead th').forEach(th => {
  headers.push(th.textContent.trim());
});

// Extract visible rows data
table.querySelectorAll('tbody tr').forEach(tr => {
  if (tr.style.display !== 'none') {
    const row = [];
    tr.querySelectorAll('td').forEach(td => {
      row.push(td.textContent.trim());
    });
    data.push(row);
  }
});

// Generate table
doc.autoTable({
  head: [headers],
```

```
        body: data,
        startY: 35,
        styles: {
          fontSize: 8,
          cellPadding: 2
        },
        headStyles: {
          fillColor: [44, 62, 80],
          textColor: 255
        }
      });

      // Save the PDF
      const timestamp = new Date().toISOString().replace(/[-:.]/g, "");
      doc.save(` ${filename}_${timestamp}.pdf` );
    }
</script>
```

RepaymentReport.cshtml

```
@model List<BankLoanProject.Models.RepaymentReportViewModel>
@{
  var role = Context.Session.GetString("Role");
  Layout = null;
}

<!-- Repayment Report Section -->
<div id="repayment" class="report-section table-container">
  <div class="section-header">
```

```
<div class="section-info">
  <span class="badge bg-success mb-2">Repayment Reports</span>
  <h4 class="mb-2">
    <i class="fas fa-receipt me-2"></i>
    Repayment Report
  </h4>
  <p class="text-muted">Monitor loan repayment schedules and payment history.</p>
</div>
<div class="export-filter-controls">
  <button class="btn btn-export" id="btnExport">
    <i class="fas fa-file-pdf"></i> Export PDF
  </button>
  <!-- Filter Dropdown -->
  <div class="dropdown">
    <button class="btn btn-filter dropdown-toggle" type="button" data-bs-toggle="dropdown">
      <i class="fas fa-filter"></i> Filter
    </button>
    <div class="dropdown-menu p-3">
      <label class="fw-bold">By Payment Date:</label>
      <input type="date" id="paymentDateFilter" class="form-control mb-2">

      <label class="fw-bold">By Status:</label>
      <select id="statusFilter" class="form-control mb-2">
        <option value="">All</option>
        <option value="Pending">Pending</option>
        <option value="Completed">Completed</option>
      </select>
```

```
            <button class="btn btn-success mt-2 w-100"
id="btnApplyFilter">Apply</button>
            <button class="btn btn-danger mt-2 w-100"
id="btnClearFilter">Clear</button>
          </div>
        </div>
      </div>
    </div>

    <!-- Table Controls - Fixed Layout -->
    <div class="table-controls">
      <!-- Left side - Rows per page -->
      <div class="rows-control">
        <label for="rowsPerPage">
          <i class="fas fa-list-ol"></i>
          Rows per page:
        </label>
        <select id="rowsPerPage" class="form-select"
onchange="changeRowsPerPage()">
          <option value="5" selected>5</option>
          <option value="10">10</option>
          <option value="15">15</option>
          <option value="20">20</option>
        </select>
      </div>

      <!-- Right side - Search -->
      <div class="search-control">
```

```
        <div class="input-group search-input-group">
          <span class="input-group-text">
            <i class="fas fa-search"></i>
          </span>
          <input type="text" class="form-control" id="searchInput" placeholder="Search repayment reports..." oninput="searchReports()">
        </div>
      </div>
    </div>

    <div class="table-wrapper">
      <table class="table" id="repaymentReportTable">
        <thead>
          <tr>
            <th><i class="fas fa-hashtag"></i> Repayment ID</th>
            @if (role == "Admin")
            {
              <th><i class="fas fa-user"></i> Customer ID</th>
              <th><i class="fas fa-user-tag"></i> Customer Name</th>
            }
            <th><i class="fas fa-hashtag"></i> Loan ID</th>
            <th><i class="fas fa-money-bill-wave"></i> Amount Due</th>
            <th><i class="fas fa-calendar-alt"></i> Due Date</th>
            <th><i class="fas fa-calendar-check"></i> Payment Date</th>
            <th><i class="fas fa-check-circle"></i> Status</th>
          </tr>
        </thead>
        <tbody class="report-table-body">
```

```
            @foreach (var report in Model)
            {
                <tr class="report-row">
                    <td><strong>@report.RepaymentId</strong></td>
                    @if (role == "Admin")
                    {
                        <td>@report.CustomerId</td>
                        <td><strong>@report.CustomerName</strong></td>
                    }
                    <td>@report.LoanId</td>
                    <td><span class="amount">@report.AmountDue.ToString("C")</span></td>
                    <td>@report.DueDate.ToString("dd/MM/yyyy")</td>
                    <td>@(report.PaymentDate.HasValue ?
report.PaymentDate.Value.ToString("dd/MM/yyyy") : "Not Paid")</td>
                    <td><span class="@(report.PaymentStatus == "Completed" ? "status-
completed" : "status-pending")">@report.PaymentStatus</span></td>
                </tr>
            }
        </tbody>
    </table>
</div>

<!-- Pagination Section -->
<div class="pagination-section">
    <div class="pagination-info" id="repaymentPaginationInfo">
        <!-- Pagination info will be generated by JavaScript -->
    </div>
    <nav aria-label="Repayment report pagination">
        <ul class="pagination" id="repaymentPaginationContainer">
```

```
            <!-- Pagination buttons will be generated by JavaScript -->
          </ul>
        </nav>
      </div>
    </div>

    <script>
      $(document).ready(function () {
        // Initialize pagination
        setTimeout(initializePagination, 100);

        // Export Table as PDF
        $("#btnExport").click(function () {
          exportToPDF('repaymentReportTable', 'Repayment_Report');
        });

        // Apply Filters
        $("#btnApplyFilter").click(function () {
          applyFilters();
        });

        // Clear Filters
        $("#btnClearFilter").click(function () {
          clearFilters();
        });
      });

      // Global variables for pagination
```

```
let currentPage = 1;
let rowsPerPage = 5;
let allRows = [];
let filteredRows = [];

function initializePagination() {
    allRows = Array.from(document.querySelectorAll('.report-row'));
    filteredRows = [...allRows];
    updateTable();
}

function applyFilters() {
    let selectedStatus = $("#statusFilter").val().toLowerCase();
    let paymentDate = $("#paymentDateFilter").val();

    filteredRows = allRows.filter(row => {
        const cells = row.querySelectorAll('td');
        const role = '@role';
        let status, rowPaymentDate;

        if (role === "Admin") {
            status = cells[7].textContent.trim().toLowerCase();
            rowPaymentDate = cells[6].textContent.trim();
        } else {
            status = cells[5].textContent.trim().toLowerCase();
            rowPaymentDate = cells[4].textContent.trim();
        }
```

```
        let showRow = true;

        if (selectedStatus && status !== selectedStatus) {
            showRow = false;
        }

        if (paymentDate && rowPaymentDate !== "Not Paid") {
            let dateParts = rowPaymentDate.split("/");
            if (dateParts.length === 3) {
                let formattedRowDate = `${dateParts[2]}-${dateParts[1]}-${dateParts[0]}`;
                if (paymentDate !== formattedRowDate) {
                    showRow = false;
                }
            }
        }

        return showRow;
    });

    currentPage = 1;
    updateTable();
}

function clearFilters() {
    $("#statusFilter").val("");
    $("#paymentDateFilter").val("");
    $("#searchInput").val("");
```

```
    filteredRows = [...allRows];
    currentPage = 1;
    updateTable();
}

function searchReports() {
    const searchTerm = document.getElementById('searchInput').value.toLowerCase();

    if (searchTerm === '') {
        filteredRows = [...allRows];
    } else {
        filteredRows = allRows.filter(row => {
            const text = row.textContent.toLowerCase();
            return text.includes(searchTerm);
        });
    }

    currentPage = 1;
    updateTable();
}

function changeRowsPerPage() {
    rowsPerPage = parseInt(document.getElementById('rowsPerPage').value);
    currentPage = 1;
    updateTable();
}

function updateTable() {
```

```
    // Hide all rows first
    allRows.forEach(row => row.style.display = 'none');

    // Calculate pagination
    const totalRows = filteredRows.length;
    const totalPages = Math.ceil(totalRows / rowsPerPage) || 1;
    const startIndex = (currentPage - 1) * rowsPerPage;
    const endIndex = Math.min(startIndex + rowsPerPage, totalRows);

    // Show current page rows
    for (let i = startIndex; i < endIndex; i++) {
      if (filteredRows[i]) {
        filteredRows[i].style.display = '';
      }
    }

    // Update pagination controls
    updatePagination(totalPages, totalRows, startIndex, endIndex);
  }

  function updatePagination(totalPages, totalRows, startIndex, endIndex) {
    const paginationContainer =
document.getElementById('repaymentPaginationContainer');
    const paginationInfo = document.getElementById('repaymentPaginationInfo');

    // Clear existing pagination
    paginationContainer.innerHTML = '';
```

```
if (totalPages <= 1) {
  const singleBtn = createPaginationButton('1', 1, false, true);
  paginationContainer.appendChild(singleBtn);
} else {
  // First page button
  const firstBtn = createPaginationButton('«', 1, currentPage === 1);
  paginationContainer.appendChild(firstBtn);

  // Previous page button
  const prevBtn = createPaginationButton('‹', currentPage - 1, currentPage === 1);
  paginationContainer.appendChild(prevBtn);

  // Page number buttons
  const maxVisiblePages = 5;
  let startPage = Math.max(1, currentPage - Math.floor(maxVisiblePages / 2));
  let endPage = Math.min(totalPages, startPage + maxVisiblePages - 1);

  if (endPage - startPage + 1 < maxVisiblePages) {
    startPage = Math.max(1, endPage - maxVisiblePages + 1);
  }

  for (let i = startPage; i <= endPage; i++) {
    const pageBtn = createPaginationButton(i, i, false, i === currentPage);
    paginationContainer.appendChild(pageBtn);
  }

  // Next page button
```

```
        const nextBtn = createPaginationButton('›', currentPage + 1, currentPage === totalPages);
        paginationContainer.appendChild(nextBtn);

        // Last page button
        const lastBtn = createPaginationButton('»', totalPages, currentPage === totalPages);
        paginationContainer.appendChild(lastBtn);
    }

    // Update pagination info
    if (totalRows > 0) {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            Showing ${startIndex + 1} to ${endIndex} of ${totalRows} entries
        `;
    } else {
        paginationInfo.innerHTML = `
            <i class="fas fa-info-circle"></i>
            No matching records found
        `;
    }
}

function createPaginationButton(text, page, disabled, active = false) {
    const li = document.createElement('li');
    li.className = `page-item ${active ? 'active' : ''} ${disabled ? 'disabled' : ''}`;

    const button = document.createElement('a');
```

```
    button.className = 'page-link';
    button.innerHTML = text;
    button.href = '#';

    if (!disabled) {
      button.onclick = (e) => {
        e.preventDefault();
        goToPage(page);
      };
    }

    li.appendChild(button);
    return li;
}

function goToPage(page) {
    currentPage = page;
    updateTable();
}

function exportToPDF(tableId, filename) {
    const { jsPDF } = window.jspdf;
    const doc = new jsPDF();

    // Add title
    doc.setFontSize(16);
    doc.text('Repayment Report', 14, 15);
```

```
// Add date
doc.setFontSize(10);
doc.text('Generated on: ' + new Date().toLocaleDateString(), 14, 25);

// Get table data
const table = document.getElementById(tableId);
const headers = [];
const data = [];

// Extract headers
table.querySelectorAll('thead th').forEach(th => {
  headers.push(th.textContent.trim());
});

// Extract visible rows data
table.querySelectorAll('tbody tr').forEach(tr => {
  if (tr.style.display !== 'none') {
    const row = [];
    tr.querySelectorAll('td').forEach(td => {
      row.push(td.textContent.trim());
    });
    data.push(row);
  }
});

// Generate table
doc.autoTable({
  head: [headers],
```

```
        body: data,
        startY: 35,
        styles: {
          fontSize: 8,
          cellPadding: 2
        },
        headStyles: {
          fillColor: [44, 62, 80],
          textColor: 255
        }
      });

      // Save the PDF
      const timestamp = new Date().toISOString().replace(/[-:.]/g, "");
      doc.save(` ${filename}_${timestamp}.pdf` );
    }
</script>
```

Report.cshtml

```
@{
  ViewData["Title"] = "Reports";
  var role = Context.Session.GetString("Role");
}

<link rel="stylesheet" href="~/css/report.css" />
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" />
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
```

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf-
autotable/3.5.31/jspdf.plugin.autotable.min.js"></script>


<div class="main-content">
  <!-- Dashboard Header -->
  <div class="dashboard-header">
    <h1>
      <i class="fas fa-chart-bar me-3"></i>
      Reports Management
    </h1>
    <p>Generate and analyze comprehensive loan reports</p>
  </div>

  <!-- Refresh Button -->
  <div class="refresh-container">
    <button type="button" class="btn-refresh" onclick="refreshData()">
      <i class="fas fa-sync-alt"></i>
      Refresh
    </button>
  </div>

  <!-- Report Type Buttons -->
  <div class="report-type-controls">
    <div class="report-type-buttons">
      <button class="btn report-btn btn-loan-report active"
onclick="showReport('loan')">
        <i class="fas fa-file-invoice-dollar me-2"></i> Loan Report
```

```
        </button>
        <button class="btn report-btn btn-repayment-report"
onclick="showReport('repayment')">
          <i class="fas fa-receipt me-2"></i> Repayment Report
        </button>
        <button class="btn report-btn btn-outstanding-report"
onclick="showReport('outstanding')">
          <i class="fas fa-money-bill-wave me-2"></i> Outstanding Report
        </button>
      </div>
    </div>

    <!-- Dynamic Report Container -->
    <div id="reportContainer">
      <!-- Reports will be loaded here -->
    </div>
</div>

<script>
    // Global variables
    let currentReportType = 'loan';

    // Initialize on page load
    document.addEventListener('DOMContentLoaded', function() {
      // Load loan report by default
      showReport('loan');
    });

    function showReport(type) {
```

```
    // Update button states
    document.querySelectorAll('.report-btn').forEach(btn =>
btn.classList.remove('active'));
    document.querySelector(`.btn-${type}-report`).classList.add('active');

    // Save current report type
    currentReportType = type;

    // Load the appropriate report
    const reportUrls = {
      'loan': '/Report/LoanReport',
      'repayment': '/Report/RepaymentReport',
      'outstanding': '/Report/OutstandingReport'
    };

    $("#reportContainer").load(reportUrls[type], function(response, status, xhr) {
      if (status === "error") {
        $("#reportContainer").html(`
          <div class="alert alert-danger text-center">
            <i class="fas fa-exclamation-triangle me-2"></i>
            Error loading ${type} report. Please try again.
          </div>
        `);
      }
    });
  }

  function refreshData() {
```

```
    const refreshBtn = document.querySelector('.btn-refresh i');
    refreshBtn.classList.add('fa-spin');

    setTimeout(() => {
      showReport(currentReportType);
      refreshBtn.classList.remove('fa-spin');
    }, 800);
  }
</script>
```

report.css

```
/* Report Management Styling - Based on Loan Product CSS */
:root {
  --primary: #2c3e50;
  --secondary: #3498db;
  --accent: #16a085;
  --light: #f8f9fa;
  --dark: #343a40;
  --success: #27ae60;
  --warning: #f39c12;
  --danger: #e74c3c;
  --info: #1abc9c;
  --table-header-bg: linear-gradient(to right, #2c3e50, #3498db);
  --pagination-border: #495057;
  --pagination-bg: #f8f9fa;
  --pagination-active-bg: #3498db;
}
```

```
body {
    background-color: #f5f7fa;
    font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
    margin: 0;
    padding: 20px 0;
}

/* Main Content */
.main-content {
    padding: 20px 0;
    min-height: 100vh;
}

/* Header Container */
.dashboard-header {
    background: linear-gradient(135deg, var(--primary), var(--secondary));
    color: white;
    border-radius: 15px;
    padding: 25px;
    margin: 25px auto;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
    text-align: center;
    max-width: 96%;
}

    .dashboard-header h1 {
        margin: 0;
        font-size: 2rem;
```

```
    font-weight: 700;
    text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 15px;
  }

  .dashboard-header p {
    margin: 10px 0 0 0;
    opacity: 0.9;
    font-size: 1.1rem;
  }

/* Refresh Button */
.refresh-container {
  max-width: 96%;
  margin: 0 auto;
  display: flex;
  justify-content: flex-end;
  margin-bottom: 20px;
}

.btn-refresh {
  background: linear-gradient(135deg, var(--accent), var(--info));
  color: white;
  border: none;
  border-radius: 8px;
```

```
    padding: 10px 20px;
    font-weight: 600;
    transition: all 0.3s;
    box-shadow: 0 4px 6px rgba(22, 160, 133, 0.2);
    display: flex;
    align-items: center;
    gap: 8px;
}

    .btn-refresh:hover {
        background: linear-gradient(135deg, var(--info), var(--accent));
        transform: translateY(-2px);
        color: white;
        box-shadow: 0 6px 12px rgba(22, 160, 133, 0.3);
    }

    .btn-refresh i {
        font-size: 1rem;
    }

/* Report Type Controls */
.report-type-controls {
    max-width: 96%;
    margin: 20px auto;
    background-color: white;
    border-radius: 12px;
    padding: 20px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
```

```
}

.report-type-buttons {
    display: flex;
    gap: 15px;
    justify-content: center;
    flex-wrap: wrap;
}

.report-btn {
    padding: 15px 25px;
    border-radius: 10px;
    font-weight: 600;
    font-size: 1rem;
    transition: all 0.3s;
    border: 2px solid transparent;
    display: flex;
    align-items: center;
    min-width: 200px;
    justify-content: center;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    position: relative;
    overflow: hidden;
}

    .report-btn::before {
        content: "";
        position: absolute;
```

```
    top: 0;
    left: -100%;
    width: 100%;
    height: 100%;
    background: linear-gradient(90deg, transparent, rgba(255, 255, 255, 0.2),
transparent);
    transition: left 0.5s;
  }

  .report-btn:hover::before {
    left: 100%;
  }

  .report-btn i {
    font-size: 1.2rem;
    margin-right: 10px;
  }

/* Improved lighter colors for report buttons */
.btn-loan-report {
  background: linear-gradient(135deg, #74b9ff, #0984e3);
  color: white;
}

  .btn-loan-report:hover,
  .btn-loan-report.active {
    background: linear-gradient(135deg, #0984e3, #0770c4);
    transform: translateY(-3px);
```

```
    box-shadow: 0 8px 15px rgba(116, 185, 255, 0.4);
    color: white;
  }

.btn-repayment-report {
  background: linear-gradient(135deg, #55efc4, #00b894);
  color: white;
}

  .btn-repayment-report:hover,
  .btn-repayment-report.active {
    background: linear-gradient(135deg, #00b894, #00a085);
    transform: translateY(-3px);
    box-shadow: 0 8px 15px rgba(85, 239, 196, 0.4);
    color: white;
  }

.btn-outstanding-report {
  background: linear-gradient(135deg, #fdcb6e, #e17055);
  color: white;
}

  .btn-outstanding-report:hover,
  .btn-outstanding-report.active {
    background: linear-gradient(135deg, #e17055, #d63031);
    transform: translateY(-3px);
    box-shadow: 0 8px 15px rgba(253, 203, 110, 0.4);
    color: white;
```

```
    }

/* Table Controls - Fixed Layout */
.table-controls {
    background-color: white;
    border-radius: 12px;
    padding: 20px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
    margin: 20px auto;
    max-width: 96%;
    display: flex;
    justify-content: space-between;
    align-items: center;
    min-height: 70px;
}

/* Left side - Rows control */
.rows-control {
    display: flex;
    align-items: center;
    gap: 10px;
    flex: 0 0 auto;
}

    .rows-control label {
        color: var(--primary);
        font-weight: 600;
        font-size: 0.95rem;
```

```
    display: flex;
    align-items: center;
    gap: 8px;
    white-space: nowrap;
  }

  .rows-control .form-select {
    min-width: 80px;
    border: 2px solid #e0e0e0;
    border-radius: 6px;
    font-size: 0.9rem;
  }

    .rows-control .form-select:focus {
      border-color: var(--secondary);
      box-shadow: 0 0 0 0.25rem rgba(52, 152, 219, 0.25);
    }

/* Right side - Search control */
.search-control {
  display: flex;
  align-items: center;
  flex: 0 0 auto;
}

.search-input-group {
  width: 300px;
}
```

```
    .search-input-group .input-group-text {
        background-color: var(--secondary);
        color: white;
        border: 2px solid var(--secondary);
    }

    .search-input-group .form-control {
        border: 2px solid #e0e0e0;
        border-left: none;
    }

        .search-input-group .form-control:focus {
            border-color: var(--secondary);
            box-shadow: 0 0 0 0.25rem rgba(52, 152, 219, 0.25);
        }

/* Table Container */
.table-container {
    background-color: white;
    border-radius: 12px;
    padding: 0;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);
    margin: 20px auto;
    max-width: 96%;
    overflow: hidden;
    border: 4px solid var(--primary);
    min-height: 600px;
```

```
    display: flex;
    flex-direction: column;
}

/* Section Header */
.section-header {
    background: linear-gradient(135deg, #ecf0f1, #bdc3c7);
    padding: 20px 25px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    border-bottom: 2px solid var(--primary);
}

.section-info h4 {
    color: var(--primary);
    margin: 0;
    font-weight: 600;
}

.section-info p {
    margin: 5px 0 0 0;
    color: #7f8c8d;
}

/* Export and Filter Controls */
.export-filter-controls {
    display: flex;
```

```
    gap: 10px;
    align-items: center;
}

/* Improved lighter colors for export and filter buttons */
.btn-export {
    background: linear-gradient(135deg, #a29bfe, #6c5ce7);
    color: white;
    border: none;
    border-radius: 8px;
    padding: 10px 20px;
    font-weight: 600;
    transition: all 0.3s;
    box-shadow: 0 4px 6px rgba(162, 155, 254, 0.3);
    display: flex;
    align-items: center;
    gap: 8px;
}

    .btn-export:hover {
        background: linear-gradient(135deg, #6c5ce7, #5f3dc4);
        transform: translateY(-2px);
        color: white;
        box-shadow: 0 6px 12px rgba(162, 155, 254, 0.4);
    }

.btn-filter {
    background: linear-gradient(135deg, #81ecec, #00cec9);
```

```
    color: white;
    border: none;
    border-radius: 8px;
    padding: 10px 20px;
    font-weight: 600;
    transition: all 0.3s;
    box-shadow: 0 4px 6px rgba(129, 236, 236, 0.3);
    display: flex;
    align-items: center;
    gap: 8px;
}

    .btn-filter:hover {
        background: linear-gradient(135deg, #00cec9, #00b894);
        transform: translateY(-2px);
        color: white;
        box-shadow: 0 6px 12px rgba(129, 236, 236, 0.4);
    }

/* Table wrapper */
.table-wrapper {
    flex: 1;
    min-height: 400px;
    display: flex;
    flex-direction: column;
}

/* Table Styling */
```

```
.table {
    margin-bottom: 0;
    border-collapse: separate;
    border-spacing: 0;
    width: 100%;
    flex: 1;
}

    .table thead {
        background: var(--table-header-bg);
    }

        .table thead th {
            background: var(--table-header-bg);
            color: white;
            font-weight: 600;
            border: 1px solid rgba(255, 255, 255, 0.2);
            border-bottom: 2px solid rgba(255, 255, 255, 0.3);
            padding: 18px 15px;
            text-transform: uppercase;
            font-size: 0.85rem;
            letter-spacing: 1px;
            vertical-align: middle;
            position: relative;
            text-align: left;
            white-space: nowrap;
        }
```

```
        .table thead th:first-child {
            border-left: none;
        }

        .table thead th:last-child {
            border-right: none;
        }

        .table thead th i {
            margin-right: 8px;
            font-size: 1rem;
        }

.table tbody td {
    padding: 18px 15px;
    border: 1px solid #e0e0e0;
    border-top: none;
    vertical-align: middle;
    word-break: break-word;
}

    .table tbody td:first-child {
        border-left: none;
    }

    .table tbody td:last-child {
        border-right: none;
    }
```

```
    .table tbody tr:hover {
        background-color: rgba(52, 152, 219, 0.05);
        transition: background-color 0.2s ease;
    }

    .table tbody tr:last-child td {
        border-bottom: none;
    }

/* Status styling */
.status-approved {
    color: var(--success);
    font-weight: 700;
}

.status-pending {
    color: var(--warning);
    font-weight: 700;
}

.status-completed {
    color: var(--success);
    font-weight: 700;
}

.status-rejected {
    color: var(--danger);
```

```
    font-weight: 700;
}

/* Amount styling */
.amount {
    font-weight: 700;
    color: var(--success);
    font-size: 1rem;
}

/* Pagination section */
.pagination-section {
    background-color: #f8f9fa;
    border-top: 1px solid #e0e0e0;
    padding: 15px 20px;
    margin-top: auto;
}

.pagination-info {
    color: var(--primary);
    font-weight: 500;
    font-size: 0.9rem;
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 8px;
    margin-bottom: 8px;
}
```

```
.pagination {
   margin: 8px 0 0 0;
   gap: 8px;
   justify-content: center;
   display: flex;
   list-style: none;
   padding: 0;
}

   .pagination .page-item {
      display: inline-block;
   }

      .pagination .page-item .page-link {
         color: #2c3e50;
         background-color: var(--pagination-bg);
         border: 3px solid #2c3e50;
         padding: 12px 18px;
         font-weight: 700;
         font-size: 1rem;
         min-width: 55px;
         min-height: 50px;
         text-align: center;
         display: flex;
         align-items: center;
         justify-content: center;
         border-radius: 8px;
```

```
    transition: all 0.3s ease;
    text-decoration: none;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    cursor: pointer;
}

.pagination .page-item.disabled .page-link {
    opacity: 0.5;
    cursor: not-allowed;
    background-color: #e9ecef;
    border-color: #6c757d;
    color: #495057;
    transform: none;
    box-shadow: 0 1px 2px rgba(0, 0, 0, 0.05);
}

    .pagination .page-item.disabled .page-link:hover {
        background-color: #e9ecef;
        border-color: #6c757d;
        color: #495057;
        transform: none;
        box-shadow: 0 1px 2px rgba(0, 0, 0, 0.05);
    }

.pagination .page-item.active .page-link {
    background-color: var(--pagination-active-bg);
    color: white;
    border-color: var(--secondary);
```

```
        transform: translateY(-3px);
        box-shadow: 0 4px 8px rgba(52, 152, 219, 0.4);
        font-weight: 800;
    }

        .pagination .page-item.active .page-link:hover {
            background-color: var(--secondary);
            color: white;
            border-color: var(--primary);
        }

    .pagination .page-item:not(.disabled):not(.active) .page-link:hover {
        background-color: #e9ecef;
        border-color: var(--secondary);
        color: var(--secondary);
        transform: translateY(-2px);
        box-shadow: 0 4px 8px rgba(52, 152, 219, 0.2);
    }

/* Responsive Design */
@media (max-width: 768px) {
  .table-controls {
    flex-direction: column;
    align-items: stretch;
    gap: 15px;
    min-height: auto;
    padding: 15px;
  }
```

```
.rows-control,
.search-control {
   justify-content: center;
}

.search-input-group {
   width: 100%;
}

.table-container {
   overflow-x: auto;
   max-width: 98%;
}

.table {
   min-width: 800px;
}

.dashboard-header {
   margin: 15px auto;
   padding: 20px;
   max-width: 98%;
}

   .dashboard-header h1 {
      font-size: 1.5rem;
   }
```

```
.report-type-buttons {
  flex-direction: column;
  gap: 10px;
}

.report-btn {
  min-width: 100%;
}

.section-header {
  flex-direction: column;
  gap: 15px;
  text-align: center;
}

.refresh-container {
  max-width: 98%;
}

.pagination {
  gap: 4px;
  flex-wrap: wrap;
}

  .pagination .page-item .page-link {
    padding: 8px 12px;
    min-width: 40px;
```

```
        min-height: 40px;
        font-size: 0.9rem;
      }
}

@media (max-width: 576px) {
   .dashboard-header h1 {
      font-size: 1.3rem;
      flex-direction: column;
      gap: 10px;
   }

   .dashboard-header p {
      font-size: 1rem;
   }

   .table thead th {
      padding: 12px 8px;
      font-size: 0.75rem;
   }

   .table tbody td {
      padding: 12px 8px;
   }

   .pagination .page-item .page-link {
      padding: 6px 10px;
      min-width: 35px;
```

```
      min-height: 35px;
      font-size: 0.8rem;
    }
}

/* Loading states */
.loading {
    opacity: 0.6;
    pointer-events: none;
}

/* Utility classes */
.d-flex {
    display: flex;
}

.justify-content-end {
    justify-content: flex-end;
}

.justify-content-between {
    justify-content: space-between;
}

.text-center {
    text-align: center;
}
```

```
.fw-bold {
    font-weight: 700;
}

.w-100 {
    width: 100%;
}
```