Bootstrap Server Select - Usage Guide

A powerful, flexible select component with server-side search for Bootstrap 5.

📦 Installation

1. Download Files

Save these files to your project:

- `bootstrap-server-select.css`
- `bootstrap-server-select.js`

2. Include Dependencies

html

```html
<!-- Bootstrap 5 CSS -->
<link href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap/5.3.0/css/bootstrap.min.css" rel="
<!-- Bootstrap Icons -->
<link href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-icons/1.10.0/font/bootstrap-icons.
<!-- Bootstrap Server Select CSS -->
<link href="path/to/bootstrap-server-select.css" rel="stylesheet">
```

3. Include Scripts

html

```html
<!-- jQuery (required) -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.7.0/jquery.min.js"></script>
<!-- Bootstrap 5 JS -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap/5.3.0/js/bootstrap.bundle.min.js"
<!-- Bootstrap Server Select JS -->
<script src="path/to/bootstrap-server-select.js"></script>
```

🚀 Quick Start

HTML Structure

html

```html
<div class="mb-3">
    <label for="userSelect" class="form-label">Select User</label>
    <input type="text" class="form-control" id="userSelect" name="user" placeholder="Search for
</div>
```

JavaScript Initialization

javascript

```javascript
$(document).ready(function() {
    $('#userSelect').bootstrapServerSelect({
        apiUrl: '/api/users/search',
        minLength: 2,
        debounceDelay: 300
    });
});
```

🔧 Configuration Options

Option	Type	Default	Description
`apiUrl`	string	`'/api/search'`	Server endpoint URL
`method`	string	`'GET'`	HTTP method
`minLength`	number	`2`	Minimum characters before search
`debounceDelay`	number	`300`	Delay in milliseconds
`paramName`	string	`'q'`	Query parameter name
`displayKey`	string	`'name'`	Object property for display text
`valueKey`	string	`'id'`	Object property for value
`subtitleKey`	string	`null`	Object property for subtitle
`placeholder`	string	`'Search...'`	Input placeholder
`loadingText`	string	`'Searching...'`	Loading message
`noResultsText`	string	`'No results found'`	No results message
`errorText`	string	`'Failed to fetch results'`	Error message
`maxResults`	number	`10`	Maximum results to show
`cache`	boolean	`true`	Enable result caching
`autoSelectFirst`	boolean	`false`	Auto-select first result

🌐 Server-Side Implementation

Expected Request

```
GET /api/users/search?q=john
```

Expected Response Format

json

```
[
    {
        "id": 1,
        "name": "John Doe",
        "email": "john@example.com"
    },
    {
        "id": 2,
        "name": "John Smith",
        "email": "john.smith@example.com"
    }
]
```

Example PHP Implementation

php

```php
<?php
// api/users/search.php
header('Content-Type: application/json');

$query = $_GET['q'] ?? '';
$query = trim($query);

if (strlen($query) < 2) {
    echo json_encode([]);
    exit;
}

// Database query (example with PDO)
$pdo = new PDO($dsn, $username, $password);
$stmt = $pdo->prepare("
    SELECT id, name, email
    FROM users
    WHERE name LIKE ? OR email LIKE ?
    LIMIT 10
");
$searchTerm = "%{$query}%";
$stmt->execute([$searchTerm, $searchTerm]);

$results = $stmt->fetchAll(PDO::FETCH_ASSOC);
echo json_encode($results);
?>
```

Example Node.js/Express Implementation

javascript

```javascript
app.get('/api/users/search', async (req, res) => {
    try {
        const query = req.query.q || '';

        if (query.length < 2) {
            return res.json([]);
        }

        const users = await User.find({
            $or: [
                { name: { $regex: query, $options: 'i' } },
                { email: { $regex: query, $options: 'i' } }
            ]
        }).limit(10);

        res.json(users);
    } catch (error) {
        res.status(500).json({ error: 'Search failed' });
    }
});
```

💡 Usage Examples

Basic Usage

javascript

```javascript
$('#userSelect').bootstrapServerSelect({
    apiUrl: '/api/users/search'
});
```

Advanced Configuration

javascript

```javascript
$('#userSelect').bootstrapServerSelect({
    apiUrl: '/api/users/search',
    minLength: 1,
    debounceDelay: 500,
    displayKey: 'full_name',
    valueKey: 'user_id',
    subtitleKey: 'email',
    maxResults: 20,
    headers: {
        'Authorization': 'Bearer ' + token
    },
    onSelect: function(value, text, item) {
        console.log('Selected:', value, text);
    }
});
```

Auto-initialization with Data Attributes

html

```html
<input type="text"
       class="form-control"
       data-toggle="server-select"
       data-api-url="/api/users/search"
       data-min-length="1"
       data-display-key="full_name"
       placeholder="Search users...">
```

Custom Template

```javascript
$('#userSelect').bootstrapServerSelect({
    apiUrl: '/api/users/search',
    template: function(item, index) {
        return `
            <div class="bs-server-select-item" data-value="${item.id}" data-text="${item.name}"
                <div class="d-flex align-items-center">
                    <img src="${item.avatar}" class="rounded-circle me-2" width="32" height="32
                    <div>
                        <div class="fw-semibold">${item.name}</div>
                        <small class="text-muted">${item.email}</small>
                    </div>
                </div>
            </div>
        `;
    }
});
```

📋 Methods

setValue(value, text)

Set the selected value and display text:

```javascript
$('#userSelect').bootstrapServerSelect('setValue', 123, 'John Doe');
```

clear()

Clear the selection:

```javascript
$('#userSelect').bootstrapServerSelect('clear');
```

refresh()

Clear cache and refresh results:

```javascript
$('#userSelect').bootstrapServerSelect('refresh');
```

destroy()

Destroy the component:

javascript

```
$('#userSelect').bootstrapServerSelect('destroy');
```

🎯 Events

Listen to Events

javascript

```
$('#userSelect')
    .on('bss:select', function(e, value, text, item) {
        console.log('Item selected:', value, text);
    })
    .on('bss:clear', function(e) {
        console.log('Selection cleared');
    })
    .on('bss:open', function(e) {
        console.log('Dropdown opened');
    })
    .on('bss:close', function(e) {
        console.log('Dropdown closed');
    })
    .on('bss:loaded', function(e, results, query) {
        console.log('Results loaded:', results.length);
    })
    .on('bss:error', function(e, xhr, status, error) {
        console.log('Search error:', error);
    });
```

🎨 Styling

Size Variants

html

```
<!-- Small -->
<input type="text" class="form-control form-control-sm" id="smallSelect">

<!-- Default -->
<input type="text" class="form-control" id="defaultSelect">

<!-- Large -->
<input type="text" class="form-control form-control-lg" id="largeSelect">
```

Custom CSS

```css
/* Custom dropdown styling */
.bs-server-select-dropdown {
    border-radius: 10px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

/* Custom item styling */
.bs-server-select-item:hover {
    background-color: #f0f8ff;
}
```

🔍 Form Integration

Getting Values

```javascript
// Get display text
const displayText = $('#userSelect').val();

// Get actual value (from hidden input)
const actualValue = $('#userSelect').siblings('input[type="hidden"]').val();
```

Form Submission

```html
<form id="myForm">
    <input type="text" class="form-control" name="user" id="userSelect">
    <button type="submit">Submit</button>
</form>

<script>
$('#myForm').on('submit', function(e) {
    const selectedValue = $('#userSelect').siblings('input[type="hidden"]').val();

    if (!selectedValue) {
        alert('Please select a user');
        e.preventDefault();
        return false;
    }

    // Form will submit with the hidden input value
});
</script>
```

🐛 Troubleshooting

Common Issues

1. **No results showing**: Check your API endpoint and response format
2. **Search not triggering**: Verify `minLength` setting
3. **Styling issues**: Ensure Bootstrap 5 CSS is loaded
4. **JavaScript errors**: Check jQuery and Bootstrap JS are loaded

Debug Mode

```javascript
$('#userSelect').bootstrapServerSelect({
    apiUrl: '/api/users/search',
    beforeSend: function(ajaxOptions, query) {
        console.log('Searching for:', query);
        console.log('AJAX options:', ajaxOptions);
    }
});
```

📱 Browser Support

- Chrome 60+

- Firefox 55+
- Safari 12+
- Edge 79+
- IE 11 (with polyfills)

🤝 Contributing

Issues and pull requests are welcome! Please ensure your code follows the existing style and includes appropriate tests.

📄 License

MIT License - feel free to use in personal and commercial projects.