# Penyesuaian Stok

## Penyesuaian Stok Observer

### Observer

Kali ini kita akan fokus pada Penyesuaian Stok, secara bawaan penyesuaian stok ini memang sudah ada crud nya, namun saat ini jika tambah stok atau kurangi, tabel produk tidak akan berubah. Nah untuk itu, di sini kita perlu bermain yang namanya model **observer**. Ketika Penyesuaian Stok ini di tambah atau dikurangi.

Untuk menggenerasi observer ini, bisa di lakukan dengan terminal pastinya. Buka terminal Kamu dan silakan jalankan perintah ini :

|  |
| --- |
| *Terminal* |
| **php artisan make:observer PenyesuaianStokObserver** |

Dengan perintah itu, kita akan diberikan file tepat pada **app/Observers/ PenyesuaianStokObserver.php**, buka file nya dan silakan buat 2 method yaitu untuk **created()** dan **deleting()** seperti berikut ini:

|  |
| --- |
| *app/Observers/ PenyesuaianStokObserver.php* |
| <?php  namespace App\Observers;  use App\Models\PenyesuaianStok;  use Illuminate\Support\Facades\DB;  class PenyesuaianStokObserver  {      public function created(PenyesuaianStok $penyesuaianStok){          $produk = $penyesuaianStok->produk;          $produk->jumlah\_stok += $penyesuaianStok->kuantitas\_disesuaikan;          $produk->save();      }      public function deleting(PenyesuaianStok $penyesuaianStok): void      {          DB::transaction(function() use ($penyesuaianStok){              $produk = $penyesuaianStok->produk;              $produk->jumlah\_stok -= $penyesuaianStok->kuantitas\_disesuaikan;              $produk->save();          });      }  } |

Apa yang dilakukan observe ini adalah, ketika kita ada penambahan / pengurangan stock, maka produk quantity akan bertambah / berkurang. Sesuai dengan yang kita tentukan dalam pembuatan stock adjustment.

### Modifikasi Model

Kamu tau, bahwa untuk membuat observe ini di trigger, kita harus memberi tahu model nya. Untuk itu, silakan buka model nya tepat di **app/Models/PenyesuaianStok.php**, dan silakan tambahkan observe nya seperti:

|  |
| --- |
| *app/Models/PenyesuaianStok.php* |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Model;  use App\Observers\PenyesuaianStokObserver;  use Illuminate\Database\Eloquent\Relations\BelongsTo;  use Illuminate\Database\Eloquent\Factories\HasFactory;  use Illuminate\Database\Eloquent\Attributes\ObservedBy;  #[ObservedBy(PenyesuaianStokObserver::class)]  class PenyesuaianStok extends Model  {      use HasFactory;      public function produk(): BelongsTo      {          return $this->belongsTo(Produk::class);      }  } |

Kemudian setelah itu, kita bisa fokus sekarang pada resource dari penyesuaian stok ini.

### Form

Kamu mungkin telah menyadari bahwa seringkali pemilihan data menggunakan elemen select bawaan. Pendekatan ini kurang mengesankan dalam hal **User Experience**. Inilah mengapa saya kerap menggunakan fitur searchable dan preload. Tidak semua elemen perlu dilengkapi dengan fitur tersebut, terutama untuk status dari suatu model yang memiliki lingkup yang terbatas. Namun, untuk produk dengan jumlah data yang besar, penggunaan fitur ini menjadi sangat penting.

Maka sekarang, silakan modifikasi bagian form tepat pada resource stock adjustment ini yang ada di app/Filament/Resources/PenyesuaianStokResource.php.

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| public static function form(Form $form): Form      {          return $form              ->schema([                  Forms\Components\Select::make('produk\_id')                      ->relationship('produk', 'id')                      ->searchable()                      ->preload()                      ->required(),                  Forms\Components\TextInput::make('kuantitas\_disesuaikan')                      ->required()                      ->label('Ubah jumlah stok (dikurang / ditambah)')                      ->numeric(),                  Forms\Components\Textarea::make('alasan')                      ->required()                      ->default('Restock.')                      ->maxLength(65535)                      ->placeholder('Alasan menyesuaiakan stok')                      ->columnSpanFull(),              ]);      } |

Kemudian, sekarang jika Kamu buat 1 stock adjustment saja, maka itu akan langsung berefek pada tabel produk kita.

### Table

Untuk table ini, tidak banyak yang berubah. Hanya saja, saya modifikasi pada bagian description dan juga bagian aksi nya. Untuk full nya kurang lebih seperti ini:

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| public static function table(Table $table): Table      {          return $table              ->defaultSort('created\_at', 'desc')              ->columns([                  Tables\Columns\TextColumn::make('produk.nama')                      ->sortable(),                  Tables\Columns\TextColumn::make('kuantitas\_disesuaikan')                      ->label('Disesuaikan')                      ->suffix(' Kuantitas')                      ->color('gray')                      ->numeric()                      ->sortable(),                  Tables\Columns\TextColumn::make('alasan')                      ->limit(50)                      ->placeholder('-'),                  Tables\Columns\TextColumn::make('created\_at')                      ->dateTime()                      ->sortable()                      ->toggleable(isToggledHiddenByDefault: true),                  Tables\Columns\TextColumn::make('updated\_at')                      ->dateTime()                      ->sortable()                      ->toggleable(isToggledHiddenByDefault: true),              ])              ->filters([                  SelectFilter::make('produk\_id')                      ->relationship('produk', 'nama')                      ->searchable()                      ->preload(),              ])              ->actions([                  Tables\Actions\EditAction::make(),                  Tables\Actions\DeleteAction::make(),              ])              ->bulkActions([                  Tables\Actions\BulkActionGroup::make([                      Tables\Actions\DeleteBulkAction::make(),                  ]),              ]);      } |

### Filter

Kita tentu perlu menambahkan yang namanya filter pada list record ini, kita tentu ingin menunjukkan semua data dari spesifik produk yang kita pilih. Tentu harusnya itu sangat mudah untuk dilakukan, silakan tambah SelectFilter pada bagian ->filters() di method table() seperti:

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| ->filters([  SelectFilter::make('produk\_id')  ->relationship('produk', 'nama')  ->searchable()  ->preload(),  ]) |

Dengan begitu, maka sekarang harusnya Kamu akan mempunyai filter tepat di kanan atas pada header tabel ini.

## Relasi Antar Produk dan Stok

### Relation Manager

Pada bagian ini, kita akan coba untuk menampilkan relasi dari produk ke penyesuaian stok. Untuk itu, kita perlu membuat yang namanya relation manager. Buka terminal Kamu dan jalankan perintah berikut:

|  |
| --- |
| *Terminal* |
| **php artisan make:filament-relation-manager ProdukResource penyesuaianStoks kuantitas\_disesuaikan** |

Seperti layaknya PesananResource sebelumnya, kita akan implementasikan hal yang sama tentu nya.

Dengan perintah di atas, maka kita akan diberikan file tepat pada **app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php**. Yang mana itu akan kita daftarkan tepat pada resource **app/Filament/Resources/ProdukResource.php**, buka file nya dan silakan modifikasi method yang namanya **getRelations()** menjadi seperti:

|  |
| --- |
| *app/Filament/Resources/ProdukResource.php* |
| public static function getRelations(): array      {          return [             App\Filament\Resources\ProdukResource\RelationManagers\PenyesuaianStoksRelationManager::class,          ];      } |

Sekarang, jika Kamu coba edit 1 produk, maka harusnya akan ada muncul tabel tepat di paling bawah, menunjukkan semua data penyesuaian stok yang telah Kamu buat.

### Reuse Table

Jika Kamu buka file **app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php**, maka harusnya akan ada 2 methods yaitu forms() dan juga table(), itu adalah representasi dari field kolom yang ada di database. Dan karena tadi waktu saat kita generate relation manager nya menggunakan kuantitas\_disesuaikan, maka yang ditunjukkan adalah kolom itu pada tabel kuantitas\_disesuaikan. Namun ingat, bahwa kita sudah mempunyai tabel tepat di resource app/Filament/Resources/PenyesuaianStokResource.php, yang mana itu bisa kita gunakan ulang sebenarnya. Untuk mengguanaknnya tentu sangat mudah sekali, kita tinggal memanggil method **table()** yang ada di resource **PenyesuaianStokResource.php** tepat di dalam table yang ada di **PenyesuaianStoksRelationManager.php**.

Maka untuk itu, silakan buka file **app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php** dan silakan edit pada bagian **table** nya menjadi seperti:

|  |
| --- |
| *app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php* |
| public function table(Table $table): Table      {          return PenyesuaianStokResource::table($table);      } |

Dengan begitu, maka kita akan mendapatkan hasil yang persis seperti yang ada di **PenyesuaianStokResource**. Namun, ada yang mesti kita tambah tepat pada resource **PenyesuaianStokResource**, kita tentu ingin menampilkan kolom produk jika memang kita sedang melihat resource Penyesuaian Stok, namun ketika kita berada di dalam **PenyesuaianStoksRelationManager.php**, tentu kita ingin menghilangkannya menggunakan method **hiddenOn()**. Maka untuk itu, silakan buka file app/Filament/Resources/PenyesuaianStokResource.php untuk memperbarui bagian tabel nya menjadi seperti:

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| public static function table(Table $table): Table      {          return $table              ->defaultSort('created\_at', 'desc')              ->columns([                  Tables\Columns\TextColumn::make('produk.nama')                      ->sortable()                      ->hiddenOn(PenyesuaianStoksRelationManager::class),                  Tables\Columns\TextColumn::make('kuantitas\_disesuaikan')                      ->label('Disesuaikan')                      ->suffix(' Kuantitas')                      ->color('gray')                      ->numeric()                      ->sortable(),                  Tables\Columns\TextColumn::make('alasan')                      ->limit(50)                      ->placeholder('-'),                  Tables\Columns\TextColumn::make('created\_at')                      ->dateTime()                      ->sortable()                      ->toggleable(isToggledHiddenByDefault: false),                  Tables\Columns\TextColumn::make('updated\_at')                      ->dateTime()                      ->sortable()                      ->toggleable(isToggledHiddenByDefault: true),              ])              ->filters([...])              ->actions([...])              ->bulkActions([...]),              ]);      } |

Seperti biasa, ketika saya tambahkan ([...]) itu berarti kode tersebut tidak saya ubah sama sekali. Dan jangan lupa untuk mengimport class PenyesuaianStoksRelationManager.

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| <?php  namespace App\Filament\Resources;  use Filament\Forms;  use Filament\Tables;  use Filament\Forms\Form;  use Filament\Tables\Table;  use App\Models\PenyesuaianStok;  use Filament\Resources\Resource;  use Filament\Tables\Filters\SelectFilter;  use App\Filament\Resources\PenyesuaianStokResource\Pages;  use App\Filament\Resources\ProdukResource\RelationManagers\PenyesuaianStoksRelationManager;  class PenyesuaianStokResource extends Resource  {  ...  } |

Jika Kamu perhatikan baik-baik, apa yang saya robah pada tabel itu hanya pada kolom nama produk nya dan filters nya, agar disembunyikan pada class relation manager. Sehingga mereka akan tetap muncul pada **PenyesuaianStokResource** namun disembunyikan pada relation manager **PenyesuaianStoksRelationManager**.

### Reuse Form

Ingat, bahwa pada kolom produk.nama kita sengaja sembunyikan, maka kita perlu juga menyembunyikan form untuk pemilihan produk nya. Buka file **PenyesuaianStokResource.php** dan silakan edit form nya menjadi seperti:

|  |
| --- |
| *app/Filament/Resources/PenyesuaianStokResource.php* |
| public static function form(Form $form): Form      {          return $form              ->schema([                  Forms\Components\Select::make('produk\_id')                      ->relationship('produk', 'nama')                      ->searchable()                      ->preload()                      ->required()                      ->hiddenOn(PenyesuaianStoksRelationManager::class),                  Forms\Components\TextInput::make('kuantitas\_disesuaikan')                      ->required()                      ->label('Ubah jumlah stok (dikurang / ditambah)')                      ->numeric(),                  Forms\Components\Textarea::make('alasan')                      ->default('Restock.')                      ->maxLength(65535)                      ->placeholder('Alasan menyesuaiakan stok')                      ->columnSpanFull(),              ]);      } |

Kemudian setelah itu, silakan buka kembali file **PenyesuaianStoksRelationManager.php** untuk memodifikasi bagian method form() nya seperti:

|  |
| --- |
| *app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php* |
| public function form(Form $form): Form      {          return PenyesuaianStokResource::form($form);      } |

sehingga sekarang pada PenyesuaianStoksRelationManager ini hanya tinggal seperti:

|  |
| --- |
| *app/Filament/Resources/ProdukResource/RelationManagers/PenyesuaianStoksRelationManager.php* |
| <?php  namespace App\Filament\Resources\ProdukResource\RelationManagers;  use App\Filament\Resources\PenyesuaianStokResource;  use Filament\Forms\Form;  use Filament\Resources\RelationManagers\RelationManager;  use Filament\Tables\Table;  class PenyesuaianStoksRelationManager extends RelationManager  {      protected static string $relationship = 'penyesuaianStoks';      public function form(Form $form): Form      {          return PenyesuaianStokResource::form($form);      }      public function table(Table $table): Table      {          return PenyesuaianStokResource::table($table);      }  } |

## Import Data

### Class Importer

Sama layaknya seperti export, kita tentu butuh class untuk importer nya. Untuk itu, silakan buka terminal Kamu dan silakan jalankan perintah ini:

|  |
| --- |
| *Terminal* |
| **php artisan make:filament-importer Produk** |

Dengan begitu, maka kita akan diberikan 1 file yaitu: **app/Filament/Imports/ProdukImporter.php**.

### Beritahu kolomnya

By default, file ini akan membuatkan kita 3 methods yaitu getColumns(), resolveRecord() dan getCompletedNotificationBody(). Dalam hal ini, kita ingin modifikasi apa yang ada di getColumns() dan juga resolveRecord(). Untuk itu, silakan ganti apa yang ada di method getColumns() menjadi seperti:

|  |
| --- |
| *app/Filament/Imports/ProdukImporter.php* |
| public static function getColumns(): array      {          return [              ImportColumn::make('gambar')                  ->rules(['max:255']),              ImportColumn::make('nama')                  ->requiredMapping()                  ->rules(['required', 'max:255']),              ImportColumn::make('sku')                  ->label('SKU')                  ->requiredMapping()                  ->rules(['required', 'max:255']),              ImportColumn::make('deskripsi')                  ->rules(['max:65535']),              ImportColumn::make('jumlah\_stok')                  ->requiredMapping()                  ->numeric()                  ->rules(['required', 'integer']),              ImportColumn::make('harga')                  ->requiredMapping()                  ->numeric()                  ->rules(['required', 'integer']),              ImportColumn::make('harga\_modal')                  ->requiredMapping()                  ->numeric()                  ->rules(['required', 'integer']),          ];      } |

Perhatikan bahwa ada 2 kolom yang hilang yaitu gambar dan juga kategori, itu sengaja kita lakukan agar pengguna juga tidak sulit untuk mencari gambar dan kita juga sebagai creator tidak sulit untuk restructure apa yang telah berjalan. Untuk itu, ketika selesai kita lakukan import, maka harusnya penggunakan akan langsung melakukan edit untuk kategori dan gambar (jika diperlukan).

### Resolver

Pada tabel produk ini jika Kamu ingat, kita ada kolom sku yang sifatnya unique. Tentu kita tidak duplicate kolom ini, oleh karena itu kita perlu menentukan apa field yang tidak memperbolehkan ini.

|  |
| --- |
| *app/Filament/Imports/ProdukImporter.php* |
| public function resolveRecord(): ?Produk      {          return Produk::firstOrNew([              'sku' => $this->data['sku'],          ]);      } |

### List Produk

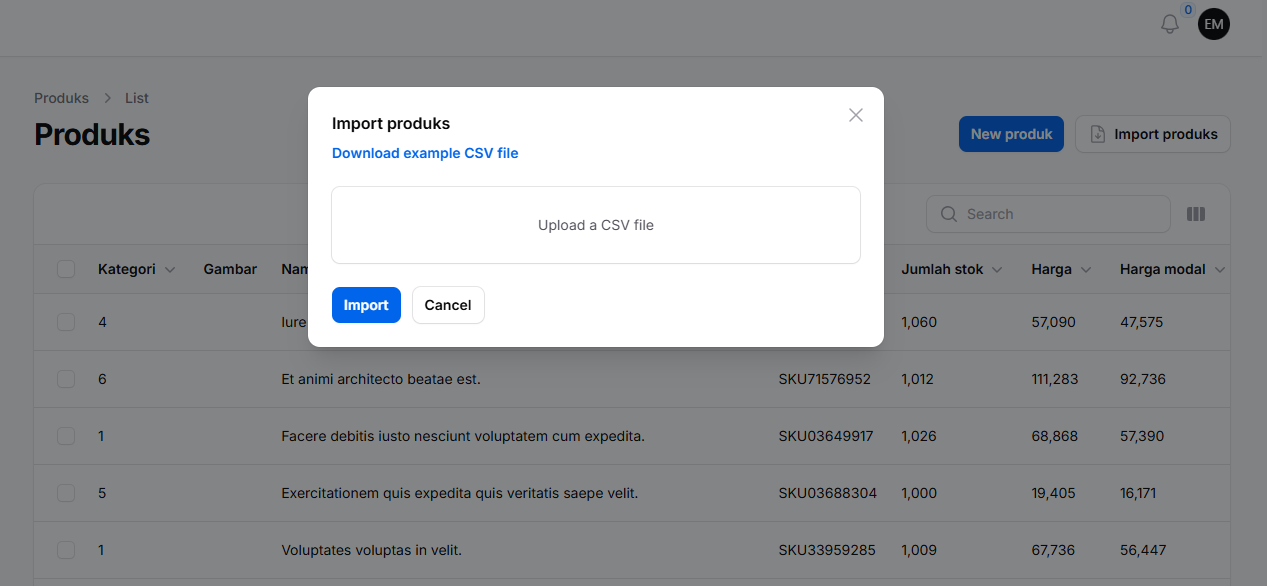
Sekarang, Kamu bisa buka file **ListProduk**, itu bisa Kamu lihat di **app/Filament/Resources/ProdukResource/Pages/ListProduk.php**, dan silakan modifikasi method getHeaderActions() seperti:

|  |
| --- |
| *app/Filament/Resources/ProdukResource/Pages/ListProduk.php* |
| <?php  namespace App\Filament\Resources\ProdukResource\Pages;  use Filament\Actions;  use App\Filament\Imports\ProdukImporter;  use Filament\Resources\Pages\ListRecords;  use App\Filament\Resources\ProdukResource;  class ListProduks extends ListRecords  {      protected static string $resource = ProdukResource::class;      protected function getHeaderActions(): array      {          return [              Actions\CreateAction::make(),              Actions\ImportAction::make()                  ->importer(ProdukImporter::class)                  ->icon('heroicon-o-document-arrow-down'),          ];      }  } |

Dengan begitu, sekarang jika Kamu refresh pada list produk di browser, maka harusnya akan ada tombol **import** tepat di samping tombol **buat**.

Sayangnya, kita tidak dapat menentukan format untuk import, jadi itu hanya bisa csv saja. Tapi itu juga tidak masalah, karena jika data dari excel, itu bisa save as csv.

Jika berhasil, maka harusnya tampilan akan seperti ini:



### Queue

Jika tidak berhasil, mungkin pada **QUEUE\_CONNECTION**Kamu menggunakan **database**, sehingga memang ini akan berjalan jika queue telah dijalankan.

|  |
| --- |
| *Terminal* |
| **php artisan queue:work** |