# Backlog – SEND To-Be

## FASE 1 – Evolução do SEND para aceitar veículos

### Descrição:

Evoluir o SEND para processar contratos de alienação de **veículos** além de consórcio, incluindo fluxo de tratamento de documentos não encontrados (24h) e integração com Kafka para eventos.

### Título:

**SEND – Fase 1: Inclusão de veículos e tratamento not found**

### História de Usuário:

**[Como]** analista de integração de gravames  
**[Quero]** que o SEND aceite e processe também contratos de alienação de veículos  
**[Para]** reduzir manualidade e riscos, ampliando cobertura da automação.

### Narrativa de negócio:

Hoje os analistas só têm processo automatizado para consórcio. Para veículos, é necessário fazer tratamento manual. Com a evolução, os veículos passam a ser aceitos pelo SEND, mantendo compatibilidade com consórcio, e eventos de documentos não encontrados são publicados em Kafka.

### Fluxo:

1. API Gravames de Veículos recebe solicitação.
2. processar-inclusao-gravame-lambda-function inicia o fluxo.
3. Payload enviado para processamento-envio-contrato-alienacao-queue.
4. preca-processar-envio-contrato-alienacao-lambda-function executa normalização.
5. Documento encontrado → base64-di4-file → envca-enviar-contrato-alienacao-lambda-function → B3-SEND/BitOne.
6. Documento não encontrado → doc-notfound-handler → fifo-doc-not-found-event (delay 2h) → di4-notfound-error-handler → dlq-not-found ou Kafka (highway-topic-veiculo-doc-notfound).
7. Documentos enviados com sucesso → Kafka (highway-topic-veiculos-docs-enviados-sucesso-B3Send).

### Impactos esperados:

* SEND deixa de ignorar veículos.
* Redução da carga manual de analistas.
* Publicação de eventos de erro em Kafka para rastreabilidade.

### Premissas de desenvolvimento:

* Usar Lambdas, SQS, SNS, Kafka (AWS managed services).
* Não quebrar fluxo de consórcio.
* Garantir compatibilidade com B3-SEND e BitOne.

### Regras de Negócio:

* **RN1:** Documentos de veículos devem ser aceitos no SEND.
* **RN2:** Documentos não encontrados em 24h → Kafka highway-topic-veiculo-doc-notfound.
* **RN3:** Documentos encontrados → enviados via B3-SEND/BitOne.
* **RN4:** Funções devem manter compatibilidade com consórcio.
* **RN5:** Logs e métricas segregam veículos x consórcio.

### Critérios de Aceite:

* CA1: Payload de veículos é aceito e processado corretamente.
* CA2: Documentos não encontrados em 24h → evento Kafka.
* CA3: Documentos encontrados são enviados via B3-SEND/BitOne.
* CA4: Logs diferenciam consórcio e veículos.

### Requisitos Funcionais:

* RF1: Evoluir preca-... e envca-... para veículos.
* RF2: Criar doc-notfound-handler e di4-notfound-error-handler.
* RF3: Implementar filas fifo-doc-not-found-event, dlq-not-found.
* RF4: Publicar eventos Kafka de erro e sucesso.

### Requisitos Não Funcionais:

* RNF1: Performance equivalente ao fluxo consórcio.
* RNF2: Confiabilidade ≥ 99,5%.
* RNF3: Observabilidade separada por domínio.
* RNF4: Segurança: logs sem PII.

## 🔹 FASE 2 – Governança via Feature Toggle e rollout seguro

### Descrição:

Incluir camada de **Feature Toggle** (Portal Manager → sync → quickconfig-cluster → cache-parametros), permitindo rollout gradual de veículos sem impactar consórcio, com fallback seguro e alta cadência (~400k execuções/dia).

### Título:

**SEND – Fase 2: Feature Toggle, Cache e Governança**

### História de Usuário:

**[Como]** analista de integração de gravames  
**[Quero]** habilitar/desabilitar o processamento de veículos via Portal Manager (Feature Toggle)  
**[Para]** controlar rollout de forma segura sem impactar o consórcio.

### Narrativa de negócio:

Hoje o SEND processa consórcio e, com a Fase 1, passa a processar veículos. Porém, é necessário controlar a ativação via Feature Toggle, garantindo rollback rápido e rollout gradual em ambientes DEV/HML/PRD.

### Fluxo:

1. Analista altera toggle no Portal Manager.
2. Serviço sincroniza-valores-cadastrados atualiza quickconfig-cluster.
3. quickconfig-cluster propaga valores para cache-parametros.
4. Lambdas (processar-inclusao-gravame, preca, envca) consultam cache antes de processar veículos.
5. Se toggle ativo → veículos processados normalmente.
6. Se toggle inativo → veículos ignorados (fluxo consórcio segue inalterado).

### Impactos esperados:

* Controle granular de rollout.
* Rollback rápido via toggle.
* Segurança para ativação em ambientes separados.

### Premissas de desenvolvimento:

* Integração Portal Manager ↔ sync ↔ quickconfig ↔ cache.
* Toggle auditável e propagação em ≤ 60s.
* Fallback: em falha, veículos ficam desativados.

### Regras de Negócio:

* **RN1:** Veículos só processados se toggle ativo.
* **RN2:** Se cache indisponível → fallback = veículos desativados.
* **RN3:** Alterações do Portal Manager refletem no cache em ≤ 60s.
* **RN4:** Toggles devem suportar granularidade (ambiente, UF, tipo de veículo).
* **RN5:** Telemetria deve diferenciar tráfego consórcio x veículos.

### Critérios de Aceite:

* CA1: Com toggle false, veículos ignorados, consórcio inalterado.
* CA2: Com toggle true, veículos processados e enviados.
* CA3: Alterações no Portal Manager propagam em ≤ 60s.
* CA4: Rollback via toggle corta tráfego de veículos em ≤ 60s.
* CA5: Latência p95 mantida ≤ baseline+5% mesmo em 400k execuções/dia.

### Requisitos Funcionais:

* RF1: Implementar integração Portal Manager ↔ sync ↔ quickconfig ↔ cache.
* RF2: Adaptar lambdas (processar-inclusao-gravame, preca, envca) para checar toggle.
* RF3: Criar governança de toggles (nomenclatura, granularidade).
* RF4: Dashboards segregados por consórcio x veículos.

### Requisitos Não Funcionais:

* RNF1: Rollout/rollback ≤ 60s.
* RNF2: Confiabilidade ≥ 99,5% mesmo sob 400k execuções/dia.
* RNF3: Segurança: toggles auditáveis e RBAC no Portal Manager.
* RNF4: Escalabilidade: cache com baixa latência e fallback seguro.

# Backlog SEND To-Be – Estrutura Jira

## EPIC 1 – FASE 1: Evolução do SEND para veículos

### Story 1.1 – Adequar preca-processar-envio-contrato-alienacao-lambda-function

* **Descrição**: Adaptar função para processar também veículos além de consórcio.
* **Tasks**:
  + Implementar branch de fluxo para veículos.
  + Normalizar payloads (chassi, RENAVAM, placa, UF).
  + Garantir idempotência e DLQ para erros.
  + Testes unitários (TDD).
  + Testes de integração (com filas).

### Story 1.2 – Adequar envca-enviar-contrato-alienacao-lambda-function

* **Descrição**: Adaptar envio para suportar veículos, B3-SEND e BitOne.
* **Tasks**:
  + Implementar mapeamento de schemas (base64-di4-file).
  + Criar switch entre consórcio x veículos.
  + Configurar autenticação separada por domínio.
  + Testes unitários e contratuais.

### Story 1.3 – Criar tratamento de documentos não encontrados

* **Descrição**: Implementar handlers para fluxos de erro de 24h.
* **Tasks**:
  + Criar doc-notfound-handler.
  + Criar di4-notfound-error-handler.
  + Criar fila fifo-doc-not-found-event (delay-exp).
  + Criar DLQ dlq-not-found.
  + Publicar evento em Kafka (highway-topic-veiculo-doc-notfound).
  + Testes end-to-end (BDD).

### Story 1.4 – Publicar eventos de sucesso em Kafka

* **Descrição**: Publicar documento encontrado e enviado com sucesso em Kafka.
* **Tasks**:
  + Configurar integração Kafka highway-topic-veiculos-docs-enviados-sucesso-B3Send.
  + Testar envio com consórcio e veículos.
  + Validar telemetria.

## EPIC 2 – FASE 2: Governança via Feature Toggle

### Story 2.1 – Implementar Portal Manager → Cache de parâmetros

* **Descrição**: Criar pipeline de governança para toggles.
* **Tasks**:
  + Implementar sincroniza-valores-cadastrados.
  + Implementar quickconfig-cluster.
  + Configurar cache-parametros (Redis/ElastiCache).
  + Criar chaves de toggle (feature.send.vehicles.enabled, granular UF/ambiente).
  + Testes de propagação (≤ 60s).

### Story 2.2 – Adaptar lambdas para leitura de toggle

* **Descrição**: Todas as funções devem condicionar execução de veículos ao toggle.
* **Tasks**:
  + Alterar processar-inclusao-gravame-lambda-function.
  + Alterar preca-....
  + Alterar envca-....
  + Implementar fallback seguro (se cache indisponível → veículos off).
  + Testes unitários e E2E (com toggle on/off).

### Story 2.3 – Rollout gradual em ambientes (DEV/HML/PRD)

* **Descrição**: Controlar rollout via toggles por ambiente.
* **Tasks**:
  + Configurar toggles separados por ambiente.
  + Criar dashboards (on/off status).
  + Testar rollout em DEV → HML → PRD.

### Story 2.4 – Observabilidade e governança

* **Descrição**: Garantir monitoramento segregado consórcio x veículos.
* **Tasks**:
  + Implementar métricas no CloudWatch/Prometheus.
  + Implementar tracing segregado por domínio.
  + Criar alarmes (toggle drift, cache miss ↑).
  + Auditar alterações de toggle (Portal Manager).

## EPIC 3 – Integração B3/BitOne (cross-phase)

(Suporta tanto Fase 1 quanto Fase 2)

### Story 3.1 – Configurar credenciais separadas (consórcio x veículos)

* Tasks:
  + Criar secrets em AWS Secrets Manager.
  + Validar conectividade B3-SEND veículos.
  + Validar conectividade BitOne veículos.

### Story 3.2 – Política de retries e rate limit

* Tasks:
  + Implementar retries exponenciais.
  + Configurar rate limit separado para veículos.
  + Testar idempotência (chave composta chassi+contrato+UF).