

FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

RM99565 – Erick Nathan Capito Pereira
RM550841 – Lucas Araujo Oliveira Silva
RM99409 – Michael José Bernardi da Silva
RM99577 – Guilherme Dias Gomes
RM550889 – Hemily Nara da Silva



Sumário

Guia de arquivos da entrega.....	3
1. Descrição do Projeto e Justificativa para o MongoDB.....	4
2. Modelo de Dados e Justificativas.....	4
2.1. Events	4
2.2. Users	4
2.3. Payments	5
2.4. Administrators.....	5
2.5. Coupons	5
2.6. Notifications	6
2.7. Polls	6
2.8. Ticket_types.....	6
2.9. Tickets.....	7
2.10. Lectures.....	7
3. Justificativa dos Modelos de Dados.....	7
4. Considerações Futuras	8
5. Análise de Performance e Escalabilidade.....	8
5.1. Expectativas de Performance	8
5.2. Escalabilidade	8
5.3. Monitoramento e Otimização	9
5.4. Considerações Finais.....	9
5.4. Considerações Finais.....	9
6. Medidas de Segurança.....	9
6.1. Controle de Acesso.....	9
6.2. Criptografia.....	9
6.3. Validação de Dados.....	9
6.4. Auditoria e Monitoramento.....	9
6.5. Backup e Recuperação.....	9
6.6. Conformidade e Melhores Práticas.....	10

Guia de arquivos da entrega

URL GITHUB: <https://github.com/geniusxp/NoSQL>

Nome do Arquivo/Pasta	Descrição
bin	Contém scripts executáveis, como de exportação e importação de dados.
Detalhes do Projeto	Pasta com detalhes e documentação e vídeo relacionada ao projeto.
ScriptsDocumentosJSON	Pasta com scripts para exportar e importar documentos JSON.
administrators.js	Arquivo JavaScript para a coleção de administradores.
coupons.js	Arquivo JavaScript para a coleção de cupons.
events.js	Arquivo JavaScript para a coleção de eventos.
lectures.js	Arquivo JavaScript para a coleção de palestras.
notifications.js	Arquivo JavaScript para a coleção de notificações.
payments.js	Arquivo JavaScript para a coleção de pagamentos.
polls.js	Arquivo JavaScript para a coleção de enquetes.
ticket_types.js	Arquivo JavaScript para a coleção de tipos de ingressos.
tickets.js	Arquivo JavaScript para a coleção de ingressos.
users.js	Arquivo JavaScript para a coleção de usuários.
CollectionExport.bat	Script em lote para exportar coleções do MongoDB para arquivos JSON.
CollectionImport.bat	Script em lote para importar arquivos JSON nas coleções do MongoDB.
public/stylesheets	Pasta contendo folhas de estilo CSS para a aplicação.
routes	Pasta com as definições de rotas da aplicação Express.
views	Pasta contendo os modelos de visualização EJS.
app.js	Arquivo principal da aplicação, que inicializa o servidor e configura as rotas.
db.js	Arquivo que lida com a conexão e lógica do banco de dados MongoDB.
package.json	Dependências e configurações do projeto Node.js.
package-lock.json	Versões exatas das dependências instaladas.
README.md	Documentação e instruções do projeto.

1. Descrição do Projeto e Justificativa para o MongoDB

O **GeniusXP** é uma plataforma de gestão de eventos projetada para simplificar processos como inscrições, pagamentos e check-in, além de aumentar o engajamento por meio de enquetes e networking. Com inteligência artificial integrada, a plataforma proporciona uma experiência personalizada, análise de sentimentos e assistência virtual, visando tornar os eventos mais impactantes e as operações de gestão mais eficientes.

Optamos pelo **MongoDB** como o banco de dados NoSQL devido à sua flexibilidade e escalabilidade. O MongoDB é ideal para armazenar dados sem uma estrutura rígida, facilitando o armazenamento de informações variadas sobre eventos, usuários, pagamentos e tickets. Sua capacidade de suportar consultas complexas e escalar horizontalmente está alinhada com o objetivo de oferecer um serviço confiável para uma base de usuários em crescimento.

2. Modelo de Dados e Justificativas

O modelo de dados no MongoDB foi elaborado para refletir as funcionalidades e requisitos do GeniusXP, preservando relacionamentos essenciais e maximizando o desempenho. Abaixo estão as coleções principais, suas estruturas, atributos e justificativas:

2.1. Coleção events

- **Descrição:** Armazena dados sobre cada evento.
- **Estrutura:**

```
_id: ObjectId('6726dc0061cd5cc5b7515edd')
id_event: 1
title: "Tech Conference 2023"
description: "Join us for a day of technology talks and networking."
date: 2023-11-15T09:00:00.000+00:00
location: "Convention Center, Downtown"
capacity: 500
registered_count: 350
status: "active"
created_at: 2023-10-01T10:00:00.000+00:00
updated_at: 2023-11-01T12:00:00.000+00:00
```

• **Justificativa:** Consolidar dados do evento, incluindo palestrantes, enquetes, tipos de ingressos e notificações, facilita consultas e reduz a necessidade de junções complexas. A estrutura aninhada permite fácil expansão e manutenção.

2.2. Coleção users

- **Descrição:** Armazena informações dos usuários.
- **Estrutura:**

```

▶ _id: ObjectId('6726ce4a3f5633ea756930b6')
  email: "davidbrown@example.com"
  password: "D@vidBr0wn"
  name: "David Brown"
  cpf: "444.333.222-11"
  birth_date: 1998-12-05T00:00:00.000+00:00
  description: "Passionate about environmental causes"
  interests: Array (3)
    0: "sustainability"
    1: "community service"
    2: "travel"
  avatar_url: "https://example.com/avatars/davidbrown.jpg"
  created_at: 2023-02-25T14:15:00.000+00:00
  updated_at: 2023-09-30T18:00:00.000+00:00

```

Justificativa: A coleção `users` contém informações relevantes para o gerenciamento de perfis e preferências. O armazenamento de votos e ingressos como subdocumentos simplifica o rastreamento de atividades e facilita a personalização.

2.3. Coleção `payments`

- **Descrição:** Armazena dados de pagamentos feitos pelos usuários.
- **Estrutura:**

```

_id: ObjectId('6726d5593f5633ea756930c1')
id_payment: 1
id_user: 123
id_ticket: 456
id_coupon: 789
payment_method: "credit_card"
amount: 100
payment_status: "paid"
payment_date: 2023-11-01T12:00:00.000+00:00
created_at: 2023-10-30T10:00:00.000+00:00
updated_at: 2023-11-01T12:00:00.000+00:00

```

Justificativa: Esta estrutura facilita o rastreamento de transações. A vinculação com `user_id` e `ticket_id` permite a recuperação eficiente de dados de pagamento por usuário ou evento.

2.4. Coleção `administrators`

- **Descrição:** Armazena dados dos administradores de eventos.
- **Estrutura:**

```

_id: ObjectId('6726d5a63f5633ea756930c6')
name: "Alice Johnson"
email: "alice.johnson@example.com"
password: "S3cur3P@ss!"
role: "Event Coordinator"
events_managed: Array (2)
last_login: 2023-10-01T09:00:00.000+00:00
created_at: 2022-01-15T10:00:00.000+00:00
updated_at: 2023-10-01T10:00:00.000+00:00
status: "Active"
permissions: Array (3)

```

Justificativa: Armazena informações dos administradores para controle de acesso. A lista `events_managed` facilita a associação dos administradores aos eventos que gerenciam.

2.5. Coleção `coupons`

- **Descrição:** Armazena informações sobre cupons de desconto.

```

_id: ObjectId('6726d5fa3f5633ea756930cb')
id_coupon : 1
code : "SUMMER2023"
description : "Summer promotion coupon"
discount_type : "percentage"
discount_value : 15
start_date : 2023-06-01T00:00:00.000+00:00
end_date : 2023-08-31T23:59:59.000+00:00
max_uses : 100
uses_count : 75
created_at : 2023-05-15T12:00:00.000+00:00
updated_at : 2023-07-20T10:30:00.000+00:00

```

- **Estrutura:**

Justificativa: Cupons são independentes de eventos específicos e podem ser aplicados em diversos contextos. A estrutura permite flexibilidade na aplicação e expiração dos cupons.

2.6. Coleção notifications

- **Descrição:** Armazena notificações enviadas aos usuários.
- **Estrutura:**

```

_id: ObjectId('6726d6653f5633ea756930d0')
id_notification : 1
id_user : 123
notification_type : "event_reminder"
title : "Upcoming event reminder"
message : "Don't forget, your event is tomorrow at 7 PM!"
read : false
created_at : 2023-11-01T08:00:00.000+00:00
updated_at : 2023-11-01T08:00:00.000+00:00

```

Justificativa: A coleção de notificações permite que a plataforma envie atualizações relevantes aos usuários, melhorando a comunicação e o engajamento. A estrutura facilita a filtragem de notificações não lidas e o gerenciamento do histórico de comunicações.

2.7. Coleção polls

- **Descrição:** Armazena dados sobre enquetes criadas para eventos.
- **Estrutura:**

```

_id: ObjectId('6726d70f61cd5cc5b7515ed8')
id_poll : 1
title : "Favorite event activity"
description : "Vote for your favorite activity at our upcoming event"
start_date : 2023-10-01T00:00:00.000+00:00
end_date : 2023-10-15T23:59:59.000+00:00
total_votes : 150
options : Array (3)
created_at : 2023-09-15T12:00:00.000+00:00
updated_at : 2023-10-16T09:00:00.000+00:00

```

Justificativa: A coleção de enquetes permite coletar feedback dos participantes sobre eventos, promovendo maior interação. A estrutura aninhada para opções facilita a contagem de votos e a análise dos resultados.

2.8. Coleção tickets

- **Descrição:** Armazena informações sobre os ingressos adquiridos pelos usuários.
- **Estrutura:**

```

_id: ObjectId('6726df7f61cd5cc5b7515ee7')
id_ticket : 1
event_id : 1
ticket_type : "Regular"
price : 100
available_quantity : 200
sold_quantity : 150
status : "available"
created_at : 2023-10-01T09:00:00.000+00:00
updated_at : 2023-11-01T12:00:00.000+00:00

```

Justificativa: A coleção tickets permite rastrear a compra e o status de cada ingresso, facilitando o gerenciamento de pessoas em eventos. A vinculação com outras coleções (users, events, ticket_types) melhora a integridade dos dados.

2.9. Coleção ticket_types

- **Descrição:** Armazena informações sobre os diferentes tipos de ingressos disponíveis para cada evento.
- **Estrutura:**

```

_id: ObjectId('6726dfdd61cd5cc5b7515eec')
id_ticket_type : 1
type : "Regular"
description : "Standard access to the event."
price : 100
max_quantity : 200
created_at : 2023-10-01T09:00:00.000+00:00
updated_at : 2023-11-01T12:00:00.000+00:00

```

Justificativa: A coleção ticket_types permite gerenciar diferentes categorias de ingressos para eventos, facilitando a venda e o controle de disponibilidade. A estrutura vinculada ao event_id permite que os tipos de ingressos sejam facilmente relacionados aos eventos específicos.

2.10. Coleção lectures

- **Descrição:** Armazena informações sobre as palestras realizadas durante os eventos.
- **Estrutura:**

```

_id: ObjectId('6726dca661cd5cc5b7515ee2')
id_lecture : 1
title : "Introduction to AI"
speaker : "Dr. Alice Smith"
date : 2023-11-10T10:00:00.000+00:00
duration : 60
description : "A comprehensive overview of artificial intelligence and its applicatio..."
event_id : 1
created_at : 2023-10-01T09:00:00.000+00:00
updated_at : 2023-11-01T12:00:00.000+00:00

```

Justificativa: A coleção "lectures" permite armazenar informações detalhadas sobre cada palestra, incluindo os palestrantes, horários, localização, materiais e os participantes. Essa estrutura facilita a organização e o gerenciamento das palestras dentro dos eventos, permitindo que os usuários acessem as informações relevantes.

A estrutura de dados em MongoDB foi projetada para centralizar informações relevantes em cada documento, refletindo diretamente a lógica do GeniusXP:

- **Eventos como Documentos:** Dados essenciais dos eventos, como palestrantes, notificações e enquetes, são armazenados diretamente no documento do evento, reduzindo a necessidade de buscas complexas e acelerando a recuperação de informações.
- **Flexibilidade para Dados de Usuários:** Os documentos users contêm detalhes de perfil e histórico de atividades, permitindo um gerenciamento eficaz das preferências e histórico dos participantes.
- **Pagamentos e Cupons Independentes:** A estrutura dos documentos payments e coupons permite rastrear transações e aplicar descontos sem dependências rígidas, melhorando a escalabilidade.
- **Escalabilidade e Eficiência:** MongoDB permite que os dados cresçam de forma escalável, suportando eventos de diferentes tamanhos e complexidade. Essa flexibilidade é fundamental para a GeniusXP, que pode adaptar o sistema conforme o aumento do volume de dados.
- **Desempenho:** As operações de leitura são otimizadas em um banco de dados NoSQL, pois muitas consultas podem ser atendidas diretamente no documento de eventos ou usuários, evitando operações de junção.

4. Considerações Futuras

À medida que a GeniusXP cresce e a demanda aumenta, consideramos as seguintes melhorias:

- **Sharding:** Com o aumento no volume de dados, o uso de sharding em MongoDB pode distribuir as coleções para melhorar a escalabilidade e o desempenho.
- **Índices:** Implementar índices em atributos frequentemente consultados, como user_id em payments e event_id em tickets, para melhorar a eficiência das consultas.
- **Migração para Microsserviços:** Caso a aplicação exija uma arquitetura mais modular, o modelo atual pode ser ajustado para suportar consultas distribuídas em uma arquitetura de microsserviços.

OBS: **Sharding** é uma técnica de particionamento de dados utilizada em bancos de dados NoSQL, como o MongoDB, para lidar com grandes volumes de dados e aumentar a escalabilidade. A ideia principal do sharding é dividir um banco de dados em partes menores, chamadas de "shards", que podem ser armazenadas em servidores diferentes. Cada shard contém uma parte dos dados e é responsável por gerenciar suas próprias operações.

5. Análise de Performance e Escalabilidade

5.1. Expectativas de Performance

A performance de um banco de dados NoSQL, como o MongoDB, é geralmente medida em termos de latência e throughput. Para o projeto GeniusXP, as seguintes expectativas podem ser definidas:

- **Latência de Consulta:** A latência para operações de leitura (consultas) deve ser minimizada. Espera-se que consultas simples retornem resultados em menos de 100 ms, enquanto consultas mais complexas, que envolvem filtragem e ordenação, podem ter uma latência de até 300 ms, dependendo da complexidade da consulta e do volume de dados.
- **Throughput:** O sistema deve ser capaz de manipular um número elevado de operações simultâneas. Espera-se que o banco de dados suporte pelo menos 100 operações por segundo para consultas e 50 operações por segundo para inserções, considerando um ambiente de produção com múltiplos usuários simultâneos.
- **Carga de Trabalho:** O sistema deve ser testado sob diferentes cenários de carga, incluindo:
 - **Carga Leve:** Menos de 10 usuários simultâneos.
 - **Carga Moderada:** Entre 10 e 50 usuários simultâneos.
 - **Carga Pesada:** Mais de 50 usuários simultâneos.

5.2. Escalabilidade

A escalabilidade do sistema GeniusXP deve ser abordada em dois aspectos: **escalabilidade vertical** e **escalabilidade horizontal**.

- **Escalabilidade Vertical:** Isso se refere ao aumento dos recursos de um único servidor (por exemplo, adicionando mais RAM ou CPU). Embora essa abordagem seja útil, ela tem limites físicos e pode se tornar cara. Para o MongoDB, uma máquina com maior capacidade de hardware pode melhorar o desempenho, mas não é uma solução a longo prazo para um crescimento significativo.
- **Escalabilidade Horizontal:** Esta é a abordagem recomendada para o GeniusXP. O MongoDB suporta sharding, que permite dividir os dados em múltiplos servidores (shards). Isso não apenas distribui a carga de trabalho, mas também aumenta a capacidade de armazenamento, permitindo que o sistema cresça de forma mais eficiente. Algumas considerações incluem:
 - Divisão de Dados:** Escolher uma chave de shard que distribua os dados de maneira uniforme entre os shards, evitando hotspots que podem causar lentidão em um shard específico.
 - Balanceamento de Carga:** O MongoDB pode redistribuir dados entre shards automaticamente para garantir que a carga esteja equilibrada.

5.3. Monitoramento e Otimização

- **Monitoramento de Performance:** É essencial implementar ferramentas de monitoramento, como o MongoDB Atlas ou o MongoDB Ops Manager, para acompanhar a performance em tempo real. Isso ajuda a identificar gargalos e permite ajustes proativos.
- **Otimização de Consultas:** Consultas devem ser otimizadas utilizando índices apropriados. Inegavelmente, índices em campos frequentemente consultados, como `user_id` em pagamentos e `event_id` em tickets, podem melhorar significativamente a eficiência das consultas.

5.4. Considerações Finais

À medida que a base de usuários da GeniusXP cresce e o volume de dados aumenta, a performance e a escalabilidade devem ser reavaliadas periodicamente. A adoção de práticas recomendadas de design e a implementação de uma estratégia de escalabilidade adequada são essenciais para garantir que o sistema continue a atender às necessidades dos usuários.

6. Medidas de Segurança

6.1. Controle de Acesso

- **Autenticação:** Implementar um mecanismo robusto de autenticação para verificar a identidade dos usuários. Isso pode incluir:
 - Senhas Fortes:** Exigir senhas complexas que incluam letras maiúsculas, minúsculas, números e caracteres especiais.
 - Autenticação de Dois Fatores (2FA):** Adicionar uma camada extra de segurança, solicitando um código enviado ao celular do usuário ou um aplicativo de autenticação.
- **Autorização:** Definir níveis de acesso com base nas funções dos usuários:
 - Usuários Comuns:** Acesso apenas às suas informações e funcionalidades básicas.
 - Administradores:** Acesso completo ao sistema, incluindo a capacidade de gerenciar eventos e usuários.

6.2. Criptografia

- **Criptografia em Repouso:** Os dados armazenados no banco de dados devem ser criptografados para proteger informações sensíveis, como senhas e dados de pagamento. O MongoDB possui suporte nativo para criptografia em repouso.
- **Criptografia em Trânsito:** Utilizar SSL/TLS para proteger dados que são transmitidos entre o cliente e o servidor, garantindo que as informações não possam ser interceptadas durante a comunicação.

6.3. Validação de Dados

- **Validação de Entrada:** Implementar validações rigorosas para dados de entrada do usuário, evitando injeções de código e outros ataques de segurança. Verificar e sanitizar todos os dados antes de armazená-los ou processá-los.
- **Limitação de Taxa:** Implementar medidas de limitação de taxa (rate limiting) para proteger a aplicação contra ataques de força bruta e negação de serviço (DoS).

6.4. Auditoria e Monitoramento

- **Logs de Auditoria:** Manter registros detalhados de todas as atividades do sistema, incluindo acessos, alterações de dados e tentativas de login. Isso ajuda na identificação de atividades suspeitas e na realização de auditorias de segurança.
- **Monitoramento de Segurança:** Utilizar ferramentas de monitoramento para detectar e alertar sobre atividades anômalas, como acessos não autorizados ou padrões de uso atípicos.

6.5. Backup e Recuperação

- **Backups Regulares:** Realizar backups regulares dos dados para garantir que informações críticas possam ser recuperadas em caso de perda de dados ou ataque cibernético.
- **Plano de Recuperação de Desastres:** Estabelecer um plano de recuperação de desastres para restaurar a operação normal após uma violação de segurança ou falha do sistema.

6.6. Conformidade e Melhores Práticas

- **Adesão a Normas de Segurança:** Garantir que a aplicação esteja em conformidade com normas de segurança, como PCI DSS (para pagamentos) e LGPD (Lei Geral de Proteção de Dados).
- **Treinamento de Funcionários:** Promover treinamentos regulares de segurança para todos os funcionários, enfatizando a importância da segurança da informação e as melhores práticas.