

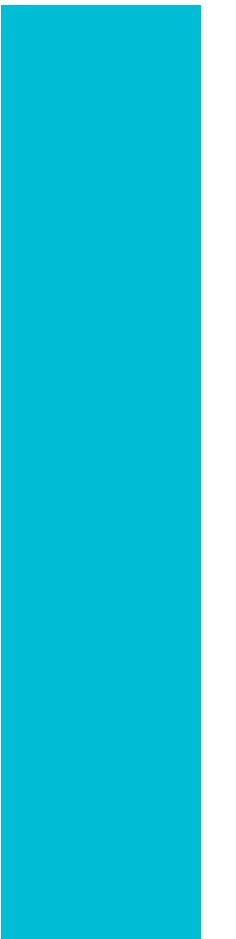
GraphQL

Techniques & Useful Tools



GraphQL Voyager

Visualize your GraphQL API

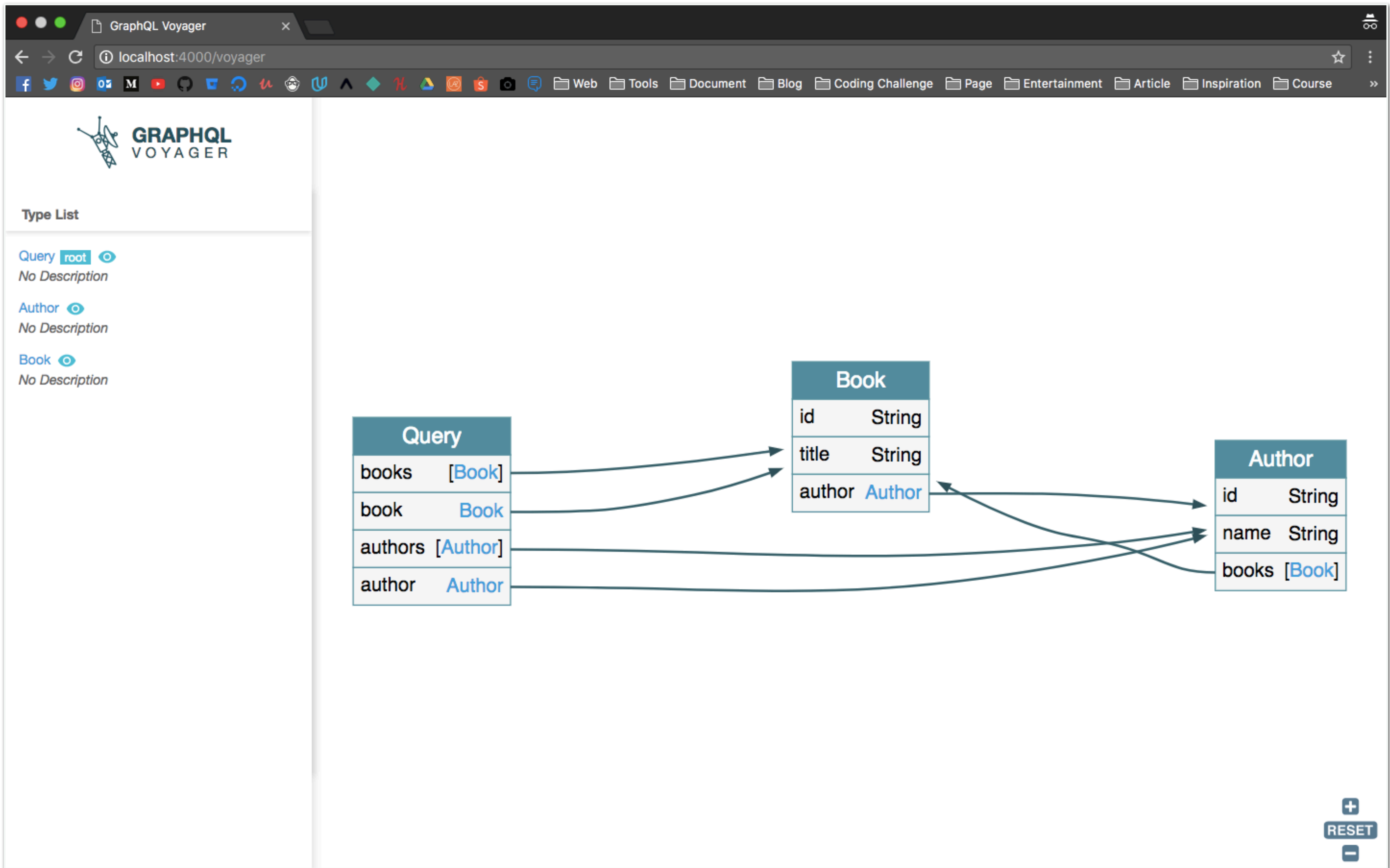




```
type Query {  
  books: [Book]  
  book(id: String): Book  
  authors: [Author]  
  author(id: String): Author  
}  
  
type Mutation {  
  createBook(title: String, authorId: String): Book  
  createAuthor(name: String): Author  
}
```



```
type Book {  
  id: String  
  title: String  
  author: Author  
}  
  
type Author {  
  id: String  
  name: String  
  books: [Book]  
}
```





Custom Directive

Validate input field using directive





```
input UserInput {  
  name: String!  
  email: String  
}
```

```
input BookInput {  
  title: String!  
  price: Int!  
}
```



```
input UserInput {  
  name: String!  
  email: String @constraint(format: "email")  
}
```

```
input BookInput {  
  title: String!  
  price: Int! @constraint(min: 0)  
}
```



Query Depth Limit

Limit unbound queries using graphql-depht-limit





```
query fetchAllBooks {  
  books {  
    author {  
      books {  
        author {  
          books {  
            author {  
              id  
            }  
          }  
        }  
      }  
    }  
  }  
}
```

Dataloader

Optimize queries via batching and caching



```
1 query {  
2   books {  
3     title  
4     author {  
5       name  
6     }  
7   }  
8 }
```



```
{  
  "data": {  
    "books": [  
      {  
        "author": {  
          "name": "J.K. Rowling"  
        },  
        "title": "Harry Potter and the Chamber of Secrets"  
      },  
      {  
        "author": {  
          "name": "J.K. Rowling"  
        },  
        "title": "Harry Potter and the Philosopher's Stone"  
      },  
      {  
        "author": {  
          "name": "J.K. Rowling"  
        },  
        "title": "Harry Potter and the Prisoner of Azkaban"  
      },  
      {  
        "author": {  
          "name": "J.K. Rowling"  
        },  
        "title": "Harry Potter and the Goblet of Fire"  
      },  
      {  
        "author": {
```

Apollo Engine

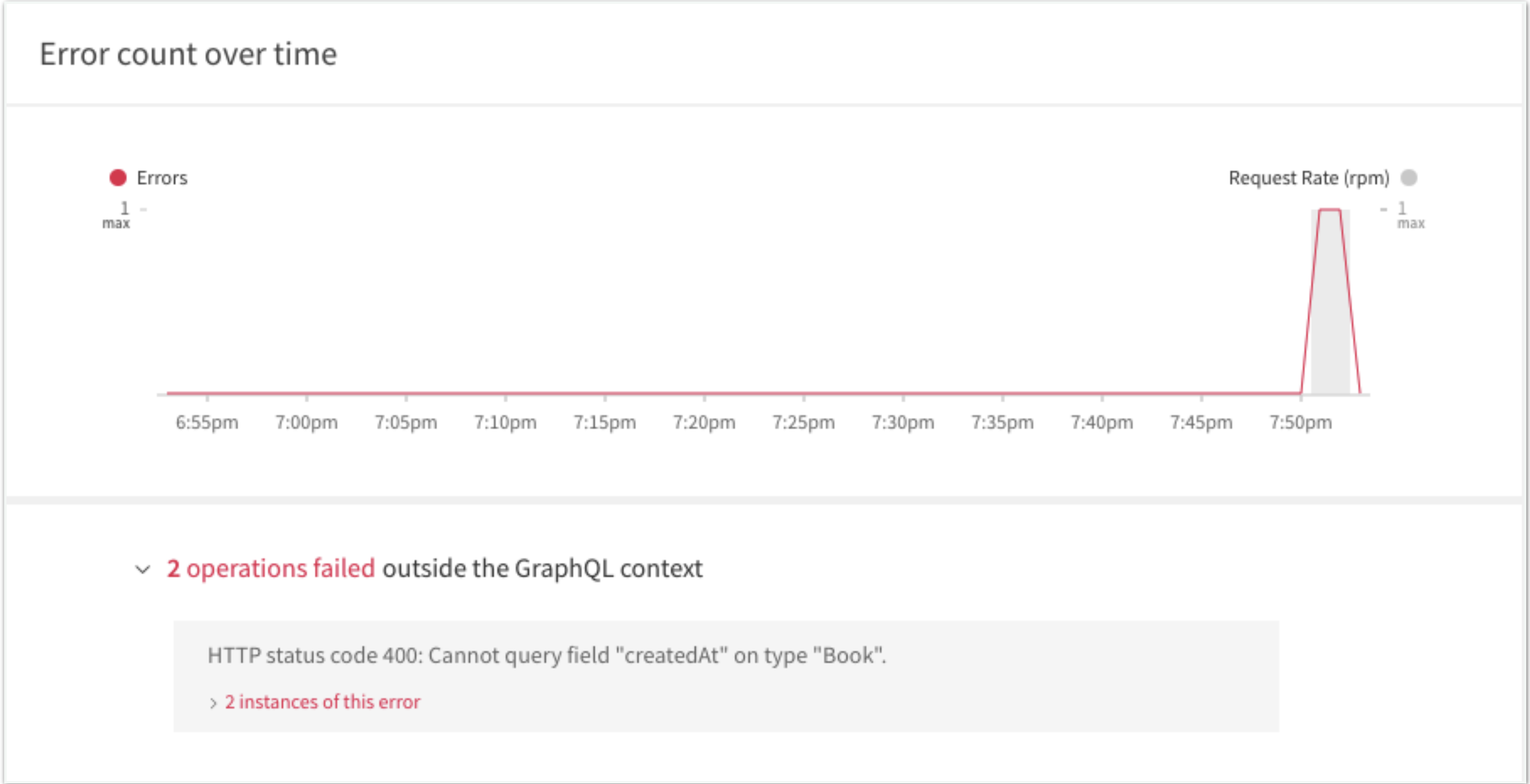
Monitoring & Caching



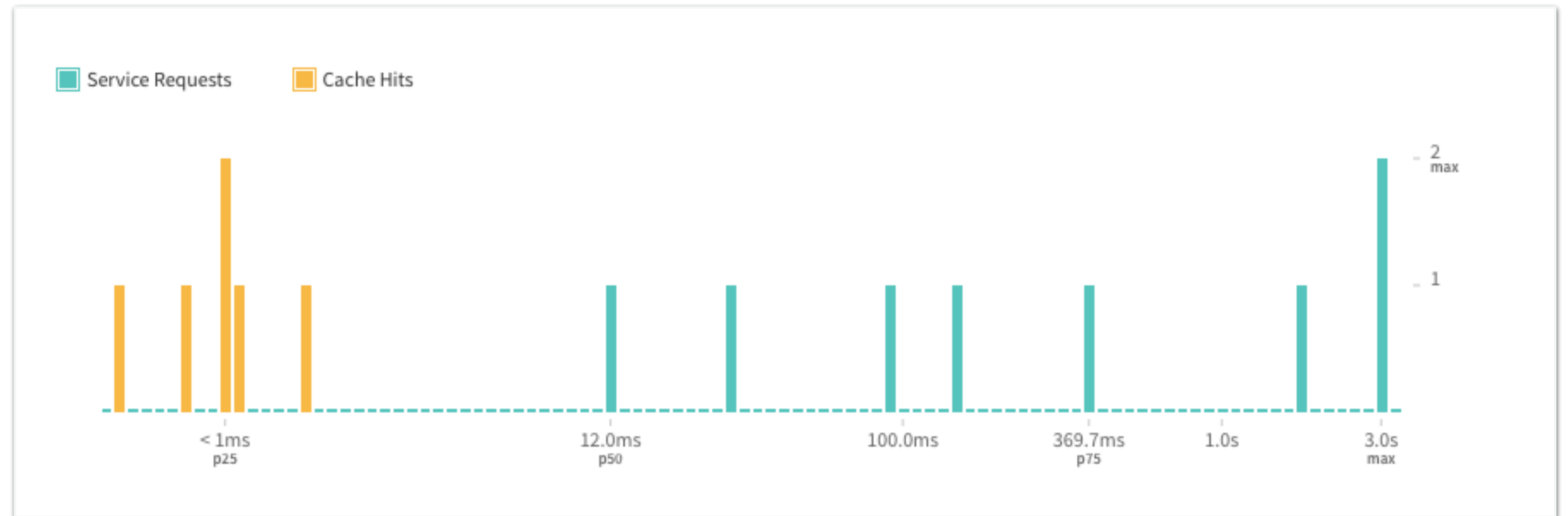
Query Execution Tracing

Execution		
Resolvers	Timing	TTL
▼ fetchBooks	1.75s	
• ▼ books:[Book]	268ms	4 min
• • ▼ books.0	,	
• • • id:String	• <1ms	
• • • title:String	• <1ms	
• • • ▼ author:Author	271ms	4 min
• • • • name:String	• <1ms	
• • ▼ books.1	,	
• • • id:String	• <1ms	
• • • title:String	• <1ms	
• • • ▼ author:Author	1.48s	4 min
• • • • name:String	• <1ms	

Error Tracking




Caching






Automatic Persisted Queries

Lighten up GraphQL queries



```
query fetchBooks {  
  books {  
    id  
    title  
    description  
    price  
    rating  
    author {  
      id  
      name  
    }  
    reviews {  
      id  
      text  
    }  
    size {  
      height  
      widht  
    }  
    updatedAt  
    createdAt
```



```
query fetchBooks {  
  books {  
    id  
    title  
    description  
    price
```

```
operationName: "fetchBooks"  
query: "query fetchBooks {↵ books {↵  id↵  title↵  author {↵    id↵    name↵  }↵ }↵}"  
variables: {}
```

```
}  
reviews {  
  id  
  text  
}  
size {  
  height  
  widht  
}  
updatedAt  
createdAt
```

```
extensions: {  
  persistedQuery: {  
    sha256Hash: "d9d6c0a2c8193815bfbc45137ea3121c11559ab0795e5651b3b4427eb39b438e"  
  }  
}  
operationName: "fetchBooks"  
variables: {}
```

```
query fetchBooks {  
  books {  
    id  
    title
```

```
    id  
    text  
  }  
  size {  
    height  
    width  
  }  
  updatedAt  
  createdAt
```

Resources

- [Sharing GraphQL techniques](#)
- [GraphQL Voyager](#)
- [GraphQL constraint directive](#)
- [GraphQL depth limit](#)
- [Dataloader](#)
- [Apollo Engine](#)
- [Automatic Persisted Queries](#)