Coverage for **app/services/request_service.py**: 85%

71 statements | 60 run | 11 missing | 0 excluded


```python
from datetime import datetime
from pathlib import Path
from app.models.request import Request
from app.schemas.request import RequestCreate, RequestRead
from app.utils.data_manager import CSVRepository

class RequestService:
    def __init__(self):
        self.repo = CSVRepository()
        self.path = Path(__file__).resolve().parents[1] / "data" / "Requests.csv"
        self.fields = ["RequestID", "UserID", "Book Title", "Author", "ISBN"]
        self.total_fields = ["ISBN", "Total Requested"]


    def __generate_next_id(self) -> int:
        """
        Generate the next RequestID number.
        """
        rows = self.repo.read_all(self.path)
        if not rows:
            return 1
        ids = [int(r["RequestID"]) for r in rows if r["RequestID"].isdigit()]
        return max(ids, default=0) + 1

    def __already_requested(self, user_id: int, isbn: str) -> bool:
        """
        Checks if this user has already requested the same book
        """
        rows = self.repo.read_all(self.path)
        return any(r["UserID"] == str(user_id) and r["ISBN"] == isbn for r in rows)

    def __decrease_count(self, isbn:str):
        path = Path(__file__).resolve().parents[1] / "data" / "Total_Requested.csv"
        rows = self.repo.read_all(path)
```

```
        for r in rows:
            if r["ISBN"] == isbn:
                new_count = int(r["Total Requested"]) - 1
                if new_count<=0: rows.remove(r)
                else:
                    r["Total Requested"] = str(new_count)
                    break

        self.repo.write_all(path, self.total_fields, rows)


    def __update_total_requested(self, isbn: str):
        path = Path(__file__).resolve().parents[1] / "data" / "Total_Requested.csv"
        rows = self.repo.read_all(path)
        found = False

        for r in rows:
            if r["ISBN"] == isbn:
                r["Total Requested"] = str(int(r["Total Requested"]) + 1)
                found = True
                break

        if not found:
            rows.append({"ISBN": isbn, "Total Requested": "1"})

        self.repo.write_all(path, self.total_fields, rows)

    def get_all_requests(self) -> list[RequestRead]:
            """
            Retrieve all requests from the requests.csv file.
            """
            rows = self.repo.read_all(self.path)
            return [RequestRead(**Request.from_dict(r).to_api_dict()) for r in rows]

    def create_request(self, user_id: int, data: RequestCreate) -> RequestRead:
        """
        Create a new book request and save it to the requests.csv file.
        """
        if self.__already_requested(user_id, data.isbn):
            raise ValueError("This user has already requested this book.")

```

```
        new_id = self.__generate_next_id()
        request = Request(
            request_id=new_id,
            user_id=user_id,
            book_title=data.book_title,
            author=data.author,
            isbn=data.isbn,
        )

        self.repo.append_row(self.path, self.fields, request.to_csv_dict())
        self.__update_total_requested(data.isbn)

        return RequestRead(**request.to_api_dict())

    def delete_request(self, request_id: int) -> bool:
        """
        Delete a specific request and reindex the remaining IDs.
        """
        rows = self.repo.read_all(self.path)
        original_count = len(rows)


        isbn_to_decrement = None
        for r in rows:
            if int(r["RequestID"]) == request_id:
                isbn_to_decrement = r["ISBN"]
                break

        # If the request ID was not found, nothing to delete
        if isbn_to_decrement is None:
            return False

        updated_rows = [r for r in rows if r["RequestID"] != str(request_id)]

        if len(updated_rows) == original_count:
            return False

        for i, row in enumerate(updated_rows, start=1):
            row["RequestID"] = str(i)

        self.repo.write_all(self.path, self.fields, updated_rows)
```

```
117         self.__decrease_count(isbn_to_decrement)
118
119         return True
```