# Documentation for the Flow Distribution Algorithm

## Overview

The flow distribution algorithm aims to fairly allocate users to astrologers while providing flexibility for top astrologers to receive more or fewer user connections. The system can handle a large volume of users and astrologers efficiently.

---

## Design Considerations

1. **Fairness**:
   Each astrologer gets an equal proportion of user connections. Top astrologers receive priority based on configuration.
2. **Scalability**:
   Designed to handle up to 3000 users and 500 astrologers per day.
3. **Flexibility**:
   Supports toggling the priority status of top astrologers.

---

## Algorithm Logic

1. Users are distributed among astrologers in a round-robin manner.
2. Top astrologers are processed first, followed by regular astrologers.
3. Each assignment increments the totalConnections attribute of the astrologer.

---

## API Endpoints

| Method | Endpoint | Description |
|---|---|---|
| POST | /flow/initialize | Initializes the astrologers' list. |
| POST | /flow/distribute | Distributes users among astrologers. |

**Example Request: /flow/initialize**

```
[
  { "id": 1, "name": "Alice", "isTopAstrologer": true },
  { "id": 2, "name": "Bob", "isTopAstrologer": false }
]
```

**Example Request: /flow/distribute**

```
[
  { "id": 1, "name": "User1" },
  { "id": 2, "name": "User2" }
]
```

---

**2. Test Cases Summary**

Tests are located in /tests/flowDistribution.test.js. Example:

- Ensures users are distributed among astrologers.
- Validates that top astrologers receive prioritized connections.

---

**3. Additional Considerations**

- **Performance**:
  Efficient loop-based assignments handle large user pools.
- **Security**:
  Use rate limiting and validation middleware in production to secure endpoints.

**Running Without Docker**

**Prerequisites**

- **Node.js** (v18 or higher)
- **npm** (Node Package Manager)

**Installation Steps**

1. cd flow-distribution-backend
2. **Install Dependencies**:
3. npm install
4. **Start the Server**:
5. node src/server.js
6. **Access the Application**:
   - The server runs on http://localhost:3000.

**API Endpoints**

**1. Initialize Astrologers**

**POST /flow/initialize**

Request Body:

```
[
  { "id": 1, "name": "Alice", "isTopAstrologer": true },
  { "id": 2, "name": "Bob", "isTopAstrologer": false }
]
```

Response:

```
{
  "message": "Astrologers initialized",
  "astrologers": [ ... ]
}
```

**2. Distribute Users**

**POST /flow/distribute**

Request Body:

```
[
  { "id": 1, "name": "User1" },
  { "id": 2, "name": "User2" }
]
```

Response:

```
{
  "message": "Flow distributed"
}
```

---

**2. Running with Docker**

**Prerequisites**

- **Docker**

- **Docker Compose**

**Installation Steps**

1. **Create the Docker Image**:
2. docker-compose build
3. **Run the Container**:
4. docker-compose up
5. **Access the Application**:
    - The server runs on [http://localhost:3000](http://localhost:3000).
6. **Stop the Container**:
7. docker-compose down

**Docker Configuration Files**

**Dockerfile**

```
FROM node:18


WORKDIR /usr/src/app


COPY package*.json ./
RUN npm install


COPY . .


EXPOSE 3000


CMD [ "node", "src/server.js" ]
```

**docker-compose.yml**

```
version: '3.8'
services:
  flow-distribution:
```

```
  build: .
  ports:
    - "3000:3000"
  volumes:
    - .:/usr/src/app
  command: node src/server.js
```

---

**Testing**

**Running Unit Tests**

1. Ensure all dependencies are installed.
2. Use the following command to run the tests:
3. npm test

Unit tests are located in /tests/flowDistribution.test.js and validate user distribution and priority handling.

---

**Additional Considerations**

- **Performance**: Designed for 2000-3000 users and 500 astrologers.
- **Scalability**: Efficient use of loops and priority filtering for top astrologers.
- **Security**: Implement additional middleware (e.g., input validation) for production use.