1. `getAccountsByUsername`

- **Описание**: Получает список идентификаторов аккаунтов, связанных с указанным именем пользователя.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, для которого нужно получить аккаунты. Не может быть пустым.
- **Выходная информация**:
  - `ResponseEntity<List<String>>`: Список идентификаторов аккаунтов, связанных с указанным пользователем. Возвращает статус 200 OK.

2. `createSettings`

- **Описание**: Создает настройки для указанного аккаунта.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта, для которого создаются настройки.
  - `settings` (Request Body): Объект `CreateSettings`, содержащий параметры для создания настроек.
- **Выходная информация**:
  - `ResponseEntity<Void>`: Возвращает статус 200 OK при успешном создании настроек.

3. `deleteSettings`

- **Описание**: Удаляет настройки по указанному идентификатору.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта, для которого удаляются настройки.
  - `settingsId` (Path Variable): Идентификатор настроек, которые нужно удалить. Не может быть null.
- **Выходная информация**:
  - `ResponseEntity<Void>`: Возвращает статус 200 OK при успешном удалении настроек.

4. `analyse` (по всем ценным бумагам акционера)

- **Описание**: Получает последние анализы для указанного акционера.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта, для которого нужно получить анализы. Не может быть пустым.
- **Выходная информация**:
  - `ResponseEntity<List<LatestAnalyse>>`: Список последних анализов для указанного аккаунта. Возвращает статус 200 OK.

5. `analyse` (по конкретной ценной бумаге)

- **Описание**: Получает последний анализ для указанного акционера и конкретной ценной бумаги.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта.
  - `securitiesId` (Path Variable): Идентификатор ценной бумаги, для которой нужно получить анализ. Не может быть пустым.
- **Выходная информация**:
  - `ResponseEntity<LatestAnalyse>`: Последний анализ для указанного аккаунта и ценной бумаги. Возвращает статус 200 OK.

6. `setRiskFree`

- **Описание**: Устанавливает новое значение безрисковой ставки для указанного аккаунта.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта, для которого устанавливается значение.
  - `newValue` (Path Variable): Новое значение безрисковой ставки. Должно быть положительным.
- **Выходная информация**:
  - `ResponseEntity<Void>`: Возвращает статус 200 OK при успешном обновлении значения или статус 204 No Content, если значение не было обновлено.

7. `setMeanBenchmark`

- **Описание**: Устанавливает новое значение среднего бенчмарка для указанного аккаунта.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, которому принадлежит аккаунт.
  - `accountId` (Path Variable): Идентификатор аккаунта, для которого устанавливается значение.
  - `newValue` (Path Variable): Новое значение среднего бенчмарка. Должно быть положительным.
- **Выходная информация**:
  - `ResponseEntity<Void>`: Возвращает статус 200 OK при успешном обновлении значения или статус 204 No Content, если значение не было обновлено.

8. `register`

- **Название**: `register`
- **Описание**: Регистрирует нового пользователя в системе.
- **Входная информация**:
  - `request` (Request Body): Объект `RegisterRequest`, содержащий данные для регистрации пользователя (например, имя пользователя, пароль, API-ключ и т.д.). Должен быть валидирован.
- **Выходная информация**:

- `ResponseEntity<RegisterResponse>`: Возвращает объект `RegisterResponse`, содержащий имя пользователя, статус успешной регистрации и информацию о том, были ли обновлены аккаунты по API-ключу. Возвращает статус 200 OK.

9. `login`

- **Название**: `login`
- **Описание**: Выполняет вход пользователя в систему.
- **Входная информация**:
  - `request` (Request Body): Объект `LoginRequest`, содержащий данные для входа (например, имя пользователя и пароль). Должен быть валидирован.
- **Выходная информация**:
  - `ResponseEntity<LoginResponse>`: Возвращает объект `LoginResponse`, содержащий информацию о результате входа (например, токен доступа). Возвращает статус 200 OK.

10. `changeToken`

- **Название**: `changeToken`
- **Описание**: Изменяет токен для указанного пользователя.
- **Входная информация**:
  - `username` (Path Variable): Имя пользователя, для которого нужно изменить токен. Не может быть пустым.
  - `token` (Path Variable): Новый токен, который нужно установить для пользователя. Не может быть пустым.
- **Выходная информация**:
  - `ResponseEntity<Void>`: Возвращает статус 200 OK при успешном изменении токена.