**[SWORLD]**

**Client Method Specification**

|  |  |
| --- | --- |
| **Ngày tạo** | [27/08/2014] |
| **Phiên bản** | [1.0] |
| **Tình trạng** | [Đề nghị] |
| **Tác giả** | [Nguyễn Công Chính, Developer] |
| **Duyệt bởi** | [] |
| **Ban hành bởi** | [] |

# LỊCH SỬ PHIÊN BẢN

| **Ngày** | **Phiên bản** | **Mô tả** | **Tác giả** |
| --- | --- | --- | --- |
|  |  | | |
| 27/08/2014 | 1.0 | Khởi tạo tài liệu | Nguyễn Công Chính |
| 25/09/2014 | 1.1 | Cập nhật eCash | Nguyễn Công Chính |
|  |  |  |  |

# NỘI DUNG

[LỊCH SỬ PHIÊN BẢN 2](#_Toc399398025)

[NỘI DUNG 3](#_Toc399398026)

[1 Quản Lý Tổ Chức 5](#_Toc399398027)

[1.1 Tổ Chức 5](#_Toc399398028)

[1.2 Tổ Chức Con 6](#_Toc399398029)

[1.3 Thành Viên 7](#_Toc399398030)

[1.4 Tổ Chức Liên Kết 8](#_Toc399398031)

[1.5 Tổ Chức Chấp Nhận Thẻ 9](#_Toc399398032)

[2 Quản Lý Khóa 11](#_Toc399398033)

[2.1 Xuất Thông Tin Public Key 11](#_Toc399398034)

[2.2 Tạo License 11](#_Toc399398035)

[2.3 Tạo Header Master Key 11](#_Toc399398036)

[2.4 Tạo Data Master Key 11](#_Toc399398037)

[2.5 Cập Nhật License Cho Thẻ 11](#_Toc399398038)

[3 Quản Lý Ứng Dụng 12](#_Toc399398039)

[4 Quản Lý Tài Khoản 14](#_Toc399398040)

[4.1 Nhóm Chức Năng 14](#_Toc399398041)

[4.2 Tài Khoản 15](#_Toc399398042)

[5 Thẻ Chíp 19](#_Toc399398043)

[5.1 Quản Lý Phát Hành 19](#_Toc399398044)

[5.2 Cấp Phát Thẻ Chíp 20](#_Toc399398045)

[5.3 Nhập Thẻ Vào Hệ Thống 21](#_Toc399398046)

[5.4 Quản Lý Thẻ Chíp 22](#_Toc399398047)

[5.5 Thông Kê 23](#_Toc399398048)

[5.6 Đọc Dữ Liệu Thẻ Chíp 23](#_Toc399398049)

[5.7 Đồng Bộ Hóa Dữ Liệu 24](#_Toc399398050)

[5.8 Xóa Dữ Liệu Trên Thẻ 24](#_Toc399398051)

[6 Chức Năng Ra/Vào 25](#_Toc399398052)

[6.1 Quản Lý Ra/Vào 25](#_Toc399398053)

[6.2 Quản Lý Đầu Số 26](#_Toc399398054)

[7 eCash – Quy Trình Nạp/Trừ Tiền Vào Thẻ 27](#_Toc399398055)

[7.1 Chứng Thực Thẻ Của SWT (Step1) 27](#_Toc399398056)

[7.2 Chứng Thực Thẻ Đối Với Partner (Nếu Có) (Step2) 27](#_Toc399398057)

[7.3 Kiểm Tra Tính Hiệu Lực Của Thẻ 27](#_Toc399398058)

[7.3.1 Transaction 27](#_Toc399398059)

[7.3.2 Client (Step3) 27](#_Toc399398060)

[7.3.3 Server 28](#_Toc399398061)

[7.4 Quy Trình Nạp Tiền Vào Thẻ 28](#_Toc399398062)

[7.4.1 Yêu Cầu Thực Hiện Nạp Tiền Vào Thẻ 28](#_Toc399398063)

[7.4.2 Cập Nhật Trạng Thái Log Nạp Tiền 31](#_Toc399398064)

[7.5 Quy Trình Trừ Tiền Trong Thẻ 32](#_Toc399398065)

[7.5.1 Yêu Cầu Thực Hiện Trừ Tiền Vào Thẻ 32](#_Toc399398066)

[7.5.2 Cập Nhật Trạng Thái Log Trừ Tiền 34](#_Toc399398067)

[7.6 Đồng Bộ Dữ Liệu 35](#_Toc399398068)

# Quản Lý Tổ Chức

## Tổ Chức

/// <summary>

/// Lấy danh sách tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách tổ chức</returns>

List<OrgCustomerDto> GetOrgList(string session, OrgFilterDto filter);

/// <summary>

/// Lấy danh sách tất cả tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <returns>Danh sách tất cả tổ chức</returns>

List<CmsOrgCustomerDto> GetAllOrgList(string session);

/// <summary>

/// Lấy thông tin của tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức</param>

/// <returns>Thông tin của tổ chức</returns>

Organization GetOrgById(string session, long OrgId);

/// <summary>

/// Thêm thông tin của tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="org">Thông tin của tổ chức</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int AddOrg(string session, Organization org);

/// <summary>

/// Cập nhật thông tin của tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="org">Thông tin của tổ chức</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateOrg(string session, Organization org);

/// <summary>

/// Xóa thông tin của tổ chức

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int RemoveOrg(string session, long orgId);

## Tổ Chức Con

/// <summary>

/// Lấy danh sách tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách tổ chức con</returns>

List<SubOrganization> GetSubOrgList(String session, long orgId, OrgFilterDto filter);

/// <summary>

/// Lấy thông tin của tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="subOrgId">Id của tổ chức con</param>

/// <returns>Thông tin của tổ chức con</returns>

SubOrganization GetSubOrgById(string session, long subOrgId);

/// <summary>

/// Thêm thông tin của tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="subOrg">Thông tin của tổ chức con</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int AddSubOrg(string session, SubOrganization subOrg);

/// <summary>

/// Cập nhật thông tin của tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="subOrg">Thông tin của tổ chức con</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateSubOrg(string session, SubOrganization subOrg);

/// <summary>

/// Xóa thông tin của tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="subOrgId">Id của tổ chức con</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int RemoveSubOrg(string session, long subOrgId);

## Thành Viên

/// <summary>

/// Lấy danh sách thành viên

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức</param>

/// <param name="subOrgId">Id của tổ chức con</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách thành viên</returns>

List<MemberCustomerDTO> GetMemberList(string session, long orgId, long subOrgId, MemberFilter filter);

/// <summary>

/// Lấy thông tin của thành viên

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="memberId">Id của thành viên</param>

/// <returns>Thông tin của thành viên</returns>

Member GetMemberById(string session, long memberId);

/// <summary>

/// Thêm thông tin của thành viên

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="member">Thông tin của thành viên</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int AddMember(string session, Member member);

/// <summary>

/// Cập nhật thông tin của thành viên

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="member">Thông tin của thành viên</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateMember(string session, Member member);

/// <summary>

/// Xóa thông tin của thành viên

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="memberId">Id của thành viên</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int RemoveMember(string session, long memberId);

/// <summary>

/// Thêm danh sách thành viên vào hệ thống

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="MemberList">Danh sách thành viên</param>

/// <returns>Danh sách thành viên không thêm được vào hệ thống</returns>

List<Member> ImportMemberData(string session, List<Member> MemberList);

## Tổ Chức Liên Kết

/// <summary>

/// Lấy danh sách Master

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <returns>Danh sách Master</returns>

List<CmsOrgCustomerDto> GetMasterList(string session);

/// <summary>

/// Lấy danh sách Partner

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách Partner</returns>

List<CmsOrgCustomerDto> GetPartnerList(string session, long masterId, OrgFilterDto filter);

/// <summary>

/// Thêm danh sách liên kết phát hành thẻ cho tổ chức được chọn

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerIdList">Danh sách Id của Partner cần liên kết phát hành thẻ</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int InsertPartnerOfMaster(string session,long masterId, List<long> partnerIdList);

/// <summary>

/// Xóa danh sách liên kết phát hành thẻ< cho tổ chức được chọn

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerIdList">Danh sách Id của Partner cần liên kết phát hành thẻ</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int DeletePartnerOfMaster(string session, long masterId, List<long> partnerIdList);

## Tổ Chức Chấp Nhận Thẻ

/// <summary>

/// Lấy danh sách tổ chức chấp nhận thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterCode">Mã xác thực của Master</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách tổ chức chấp nhận thẻ</returns>

List<CmsOrgCustomerDto> GetPartnerAcquirerList(string session, string masterCode, OrgFilterDto filter);

/// <summary>

/// Thêm danh sách chấp nhận thẻ cho tổ chức được chọn

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerIdList">Danh sách Id của Partner cần chấp nhận thẻ</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int InsertOrgAcquirer(string session, long masterId, List<long> partnerIdList);

/// <summary>

/// Xóa danh sách chấp nhận thẻ cho tổ chức được chọn

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterCode">Mã xác thực</param>

/// <param name="partnerIdList">Danh sách Id của Partner cần chấp nhận thẻ</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int DeleteOrgAcquirer(string session, string masterCode, List<long> partnerIdList);

# Quản Lý Khóa

## Xuất Thông Tin Public Key

Chức năng xuất thông tin Public Key dùng để cho API chứng thực thẻ của hệ thống sWorld.

Note: Chức năng này chưa làm.

## Tạo License

Chức năng này để tạo License mới cho hệ thống.

Note: Chức năng này chưa làm.

## Tạo Header Master Key

Chức năng này để tạo Header Master Key mới cho hệ thống.

Note: Chức năng này chưa làm.

## Tạo Data Master Key

Chức năng này để tạo Data Master Key mới cho hệ thống.

Note: Chức năng này chưa làm

## Cập Nhật License Cho Thẻ

Khi tạo mới License cho hệ thống thì cần phải cập nhật License mới cho thẻ.

Note: Chức năng này chưa làm

# Quản Lý Ứng Dụng

/// <summary>

/// Lấy danh sách ứng dụng

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="orgId">Tổ chức</param>

/// <param name="subOrgId">Tổ chức con</param>

/// <returns>Danh sách ứng dụng</returns>

List<App> GetAppDataList(string session,long orgId, long subOrgId);

/// <summary>

/// Lấy thông tin của ứng dụng

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="appId">Id của ứng dụng</param>

/// <returns>Thông tin của ứng dụng</returns>

App GetAppById(string session, long appId);

/// <summary>

/// Thêm ứng dụng

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="app">Thông tin của ứng dụng</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int InsertApp(string session, App app);

/// <summary>

/// Cập nhật ứng dụng

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="app">Thông tin của ứng dụng</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateApp(string session, App app);

/// <summary>

/// Xóa thông tin của ứng dụngs

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="appId">Id của ứng dụng</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int DeleteApp(string session, long appId);

# Quản Lý Tài Khoản

## Nhóm Chức Năng

/// <summary>

/// Lấy thông tin của nhóm chức năng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="groupId">Id của nhóm chức năng</param>

/// <returns>Thông tin của nhóm chức năng</returns>

GroupCustomerDto GetGroupById(string session, long groupId);

/// <summary>

/// Lấy danh sách nhóm chức năng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách nhóm chức năng</returns>

List<GroupDto> GetGroupList(string session, GroupFilterDto filter);

/// <summary>

/// Thêm nhóm chức năng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="group">Thông tin của nhóm chức năng</param>

/// <returns>Thông tin nhóm chức năng</returns>

GroupCustomerDto AddGroup(string session, GroupCustomerDto group);

/// <summary>

/// Cập nhật nhóm chức năng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="group">Thông tin của nhóm chức năng</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateGroup(string session, GroupCustomerDto group);

/// <summary>

/// Xóa thông tin nhóm chức năng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="groupId">Id của nhóm chức năng</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> RemoveGroups(string session, long groupId);

## Tài Khoản

/// <summary>

/// Hàm đăng nhập dựa vào username và password

/// </summary>

/// <param name="userName">Tên tài khoản</param>

/// <param name="password">Mật khẩu</param>

/// <returns>Mã session nếu đăng nhập thành công</returns>

SessionDTO Login(string userName, string password);

/// <summary>

/// Đăng xuất dựa vào mã session

/// </summary>

/// <param name="sessionId">Mã session của lần đăng nhập thành công trước đó</param>

void Logout(string session);

/// <summary>

/// Lấy danh sách tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách tài khoản</returns>

List<UserSworld> GetUserList(string session, UserFilterDto filter);

/// <summary>

/// Lấy thông tin của tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userId">Id của tài khoản</param>

/// <returns>Thông tin tài khoản</returns>

UserSworld GetUserById(string session, long userId);

/// <summary>

/// Thêm thông tin tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="user">Thông tin tài khoản</param>

/// <returns>Thông tin tài khoản</returns>

UserSworld AddUser(string session, UserSworld user);

/// <summary>

/// Cập nhật thông tin tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="user">Thông tin tài khoản</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateUser(string session, UserSworld user);

/// <summary>

/// Chuyển tài khoản sang nhóm chức năng khác

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userId">Id của tài khoản</param>

/// <param name="newGroupId">Id của nhóm chức năng mới</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int ChangeUserGroup(string session, long userId, long newGroupId);

/// <summary>

/// Đổi mật khẩu của tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="oldPassword">Mật khẩu cũ</param>

/// <param name="newPassword">Mật khẩu mới</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int ChangePassword(string session,string oldPassword, string newPassword);

/// <summary>

/// Cập nhật mật khẩu mới của tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userId">Id của tài khoản</param>

/// <param name="newPassword">Mật khẩu mới</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int ResetPassword(string session, long userId, string newPassword);

/// <summary>

/// Cập nhật tình trạng khóa cho danh sách tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userIds">Danh sách Id của tài khoản</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> LockUsers(string session, long[] userIds);

/// <summary>

/// Cập nhật tình trạng mở khóa cho danh sách tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userIds">Danh sách Id của tài khoản</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> UnLockUsers(string session, long[] userIds);

/// <summary>

/// Xóa danh sách tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userIds">Danh sách Id của tài khoản</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> RemoveUsers(string session, long[] userIds);

/// <summary>

/// Lấy danh sách lịch sử đăng nhập

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách lịch sử đăng nhập</returns>

List<LoginHistoryDTO> GetLoginHistoryList(string session, LoginHistoryFilterDto filter);

/// <summary>

/// Lấy danh sách chức năng của tài khoản được sử dụng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userId">Id của tài khoản</param>

/// <returns>Danh sách chức năng của tài khoản được sử dụng</returns>

List<PolicySworld> GetPermissionList(string session, long userId);

/// <summary>

/// Lấy danh sách tất cả chức năng

/// </summary>

/// <returns>Danh sách tất cả chức năng</returns>

List<PolicySworld> GetAllPermissionList();

/// <summary>

/// Thêm thông tin tài khoản

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="userName">Tên tài khoản</param>

/// <param name="password">Mật khẩu</param>

/// <param name="groupId">Id của nhóm chức năng</param>

/// <param name="teacherId"></param>

/// <returns>Thông tin tài khoản</returns>

UserSworld AddUser(string session, string userName, string password, long groupId, long teacherId);

/// <summary>

/// chưa được sử dụng

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="skip"></param>

/// <param name="take"></param>

/// <returns></returns>

List<LoginHistoryDTO> GetLastLoginHistoryList(string session, int skip, int take);

# Thẻ Chíp

## Quản Lý Phát Hành

/// <summary>

/// Lấy danh sách thành viên đã được cá thẻ hóa

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức phát hành</param>

/// <param name="subOrgId">Id của tổ chức con</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách thành viên đã được cá thẻ hóa</returns>

List<MemberCustomerDTO> GetMemberPersoList(string session, long orgId, long subOrgId, PersoChipFilter filter);

/// <summary>

/// Lấy danh sách tổ chức con

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="orgId">Id của tổ chức</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách tổ chức con</returns>

List<SubOrgCustomerDTO> GetSubOrgList(string session, long orgId, SubOrgFilterDto filter);

/// <summary>

/// Cập nhật tình trạng hủy của lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="ChipPersoIds">Danh sách Id của lượt phát hành</param>

/// <param name="cancelReason">Ghi chú</param>

/// <returns></returns>

List<MethodResultDto> CancelPersoes(string session, long[] ChipPersoIds, string cancelReason);

/// <summary>

/// Cập nhật tình trạng khóa của lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="ChipPersoIds">Danh sách Id của lượt phát hành</param>

/// <param name="lockReason">Ghi chú</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> LockPersoes(string session, long[] ChipPersoIds, string lockReason);

/// <summary>

/// Cập nhật tình trạng mở khóa của lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="ChipPersoIds">Danh sách Id của lượt phát hành</param>

/// <param name="unlockReason">Ghi chú</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> UnLockPersoes(string session, long[] ChipPersoIds, string unlockReason);

/// <summary>

/// Cập nhật ngày hết hạn của lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="ChipPersoIds">Danh sách Id của lượt phát hành</param>

/// <param name="expirationDate">Ghi chú</param>

/// <returns>Danh sách kết quả đã được xử lý</returns>

List<MethodResultDto> ExtendPerso(string session, long[] ChipPersoIds, string expirationDate);

/// <summary>

/// Cập nhật tình trạng của lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="status">Tình trạng lượt phát hành</param>

/// <param name="Reason">Ghi chú</param>

/// <param name="ChipPersoIds">Danh sách Id của lượt phát hành</param>

/// <returns>Danh sách lượt phát hành</returns>

List<MemberCustomerDTO> GetChangeStatus(string session, byte status, string Reason, List<long> ChipPersoIds);

## Cấp Phát Thẻ Chíp

/// <summary>

/// Lấy thông tin dữ liệu cần ghi vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="memberId">Id của Master</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="cardType">Loại thẻ</param>

/// <param name="sectorStart">Sector bắt đầu ghi ứng dụng</param>

/// <param name="AppIds">Danh sách Id cần ghi ứng dụng</param>

/// <returns>Thông tin dữ liệu cần ghi vào thẻ</returns>

DataToWriteCardDTO CheckAndGetAppDataToPersoCard(string session, long memberId, string serialNumber, int cardType, byte sectorStart, List<long> AppIds);

/// <summary>

/// Lưu thông tin lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="memberId">Id của thành viên</param>

/// <param name="serialNumberHex">Serial number của thẻ</param>

/// <param name="AppIds">Danh sách Id của ứng dụng</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int PersoCardChip(string session, long memberId, string serialNumberHex, List<long> AppIds);

## Nhập Thẻ Vào Hệ Thống

/// <summary>

/// Lấy thông tin để ghi License Master vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="id">Id của Master</param>

/// <param name="serialnumbex">Serial number của thẻ</param>

/// <param name="cardtype">Loại thẻ</param>

/// <param name="start">Sector bắt đầu ghi License</param>

/// <param name="stop">Sector kết thuc ghi License</param>

/// <returns>Dữ liệu cần ghi vào thẻ</returns>

ResultCheckCardDTO CheckAndGetMasterDataToImportCard(string session, long id, string serialnumbex, int cardtype, byte start, byte stop);

/// <summary>

/// Cập nhật thông tin của thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerId">Id của Partner</param>

/// <param name="serialnumbex">Serial number của thẻ</param>

/// <param name="cardtype">Loại thẻ</param>

/// <param name="status"> Tình trạng quyền sử dụng thẻ

/// 100: CARD\_HAS\_MASTER\_READED\_ONLY (chỉ được đọc)

/// 101: CARD\_HAS\_MASTER\_WRITED\_ONLY (Chỉ được ghi)

/// </param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateDataForCardBySerialAndMasterId(string session, long masterId, long partnerId, string serialnumber, int cardtype, int status);

/// <summary>

/// Lấy thông tin để ghi License Partner vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="id">Id của Partner</param>

/// <param name="serialnumbex">Serial number của thẻ</param>

/// <param name="cardtype">Loại thẻ</param>

/// <param name="start">Sector bắt đầu ghi License</param>

/// <param name="stop">Sector kết thuc ghi License</param>

/// <returns>Dữ liệu cần ghi vào thẻ</returns>

ResultCheckCardDTO CheckAndGetPartnerDataToImportCard(string session, long id, string serialnumbex, int cardtype, byte start, byte stop);

/// <summary>

/// Cập nhật thông tin của thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="id">Id của Partner</param>

/// <param name="serialnumber">Serial number của thẻ</param>

/// <param name="cardtype">Loại thẻ</param>

/// <param name="status"> Tình trạng quyền sử dụng thẻ

/// 103: CARD\_HAS\_MASTER\_PARTNER\_WRITE (chỉ được ghi)

/// 102: CARD\_HAS\_MASTER\_PARTNER\_READED (Chỉ được đọc)

/// </param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateDataForCardBySerialAndPartnerId(string session, long id, string serialnumber, int cardtype, int status);

## Quản Lý Thẻ Chíp

/// <summary>

/// Lấy danh sách thẻ chip

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerId">Id của Parner</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>danh sách thẻ chip</returns>

List<CardChipDto> GetCardChipList(string session, long masterId, long partnerId, CardFilterDto filter);

/// <summary>

/// Cập nhật danh sách thẻ bị mất

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="CardChipIds">Id của thẻ trong hệ thống(List)</param>

/// <returns></returns>

List<MethodResultDto> MarkBrokenCards(string session, long[] CardChipIds);

/// <summary>

/// Cập nhật danh sách thẻ hủy đánh dấu hư

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="CardChipIds">Id của thẻ trong hệ thống(List)</param>

/// <returns></returns>

List<MethodResultDto> MarkLostCards(string session, long[] CardChipIds);

/// <summary>

/// Cập nhật danh sách thẻ bị mất

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="CardChipIds">Id của thẻ trong hệ thống(List)</param>

/// <returns></returns>

List<MethodResultDto> UnMarkBrokenCards(string session, long[] CardChipIds);

/// <summary>

/// Cập nhật danh sách thẻ hủy đánh dấu mất

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="CardChipIds">Id của thẻ trong hệ thống(List)</param>

/// <returns></returns>

List<MethodResultDto> UnMarkLostCards(string session, long[] CardChipIds);

## Thông Kê

/// <summary>

/// Lấy thông tin thông kê tinh trạng của thẻ chip

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="masterId">Id của Master</param>

/// <param name="partnerId">Id của Partner</param>

/// <returns></returns>

List<CardStatisticsData> StatisticCardChipByStatus(string session, long masterId, long partnerId);

/// <summary>

/// Lấy thông tin thông kê lượt phát hành thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <returns>thông tin thống kê lượt phát hành</returns>

List<CardStatisticsData> StatisticCardChipByPersoStatus(string session);

## Đọc Dữ Liệu Thẻ Chíp

/// <summary>

/// Lấy thông tin dữ liệu của thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="cardType">Loại thẻ</param>

/// <param name="memberData">chuỗi dữ liệu đã được ghi vào thẻ</param>

/// <returns>Thông tin dữ liệu của thẻ</returns>

DataForReadCard GetDataToReadCard(string session, string serialNumber, int cardType, string memberData);

/// <summary>

/// Lấy danh sách keyB của sector để đọc thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialnumbex">Serial number của thẻ</param>

/// <param name="cardType">Loại thẻ</param>

/// <param name="list">Danh sách sector cần lấy keyB</param>

/// <returns>Thông tin của keyB của sector cần đọc thẻ</returns>

DataToReadCardDTO GetKeyForReadCard(string session, string serialNumber, int cardType, List<int> list);

## Đồng Bộ Hóa Dữ Liệu

/// <summary>

/// Lấy thông tin dữ liệu cần cập nhật vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="sectorStart"></param>

/// <returns>Thông tin dữ liệu cần cập nhật vào thẻ</returns>

DataToWriteCardDTO GetDataToUpdateCard(string session, string serialNumber, byte sectorStart);

/// <summary>

/// Cập nhật thông tin lượt phát hành

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="lastUpdateDate">Ngày cập nhật lượt phát hành</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateMemberAppOfPerso(string session, string serialNumber, string lastUpdateDate);

## Xóa Dữ Liệu Trên Thẻ

/// <summary>

/// Lấy danh sách KeyB để xóa dữ liệu trên thẻ và xóa lượt phát hành thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="cardType">Loại thẻ</param>

/// <param name="sectorStart">Sector bắt đầu xóa dữ liệu</param>

/// <param name="sectorStop">Sector kết thúc xóa dữ liệu</param>

/// <returns>Thông tin xóa dữ liệu trên thẻ</returns>

DataToWriteCardDTO CheckAndGetAppDataToClearCard(string session, string serialNumber, int cardType, byte sectorStart, byte sectorStop);

/// <summary>

/// Xóa lượt phát hành của thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int ClearCardData(string session, string serialNumber);

# Chức Năng Ra/Vào

## Quản Lý Ra/Vào

/// <summary>

/// Thêm thông tin một lượt ra/vào cửa

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="serialNumber">serial number của thẻ</param>

/// <returns>Thông tin lượt ra/vào</returns>

Attendance AddAttendance(string session, string serialNumber);

/// <summary>

/// Cập nhật thông tin một lượt ra/vào

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="attendance">Thông tin một lượt ra/vào</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int UpdateAttendance(string session, Attendance attendance);

/// <summary>

/// Cập nhật thông tin lượt ra/vào

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="attendanceId">Id của lượt ra/vào</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL,

/// 3: OKIE

/// </returns>

int RemoveAttendance(string session, long attendanceId);

/// <summary>

/// Lấy danh sách lượt ra/vào theo filter

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="filter">filter cần lọc</param>

/// <returns>Danh sách lượt ra/vào</returns>

List<Attendance> GetAttendanceList(string session, AttendanceFilterDto filter);

/// <summary>

/// Lấy danh sách lượt ra/vào theo member

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="memberId">Id của thành viên</param>

/// <returns>Danh sách lượt ra/vào</returns>

List<Attendance> GetAttendanceList(string session, long memberId);

/// <summary>

/// Lấy danh sách lượt ra/vào theo member và ngày ra

/// </summary>

/// <param name="session">Session của User đăng nhập</param>

/// <param name="memberId">Id của thành viên</param>

/// <param name="dateOut">Ngày ra</param>

/// <returns>Danh sách lượt ra/vào</returns>

List<Attendance> GetAttendanceList(string session, long memberId, string dateOut);

## Quản Lý Đầu Số

Note: Chức năng chưa làm.

# eCash – Quy Trình Nạp/Trừ Tiền Vào Thẻ

## Chứng Thực Thẻ Của SWT (Step1)

Chứng thực thẻ của SWT được thực hiện bên Client dựa vào các dữ liệu sau:

* Key Public trong file Config của hệ thống
* Dùng KeyA để đọc License trong thẻ từ sector 0 đến sector 2
* Thư viện PCSC Reader để đọc Serial Number của thẻ

Tiếp theo, sử dụng Key Public để giải mã License theo thuật toán RSA, đối chiếu kết quả giải mã với Serial Number. Nếu đúng thì thẻ thuộc hệ thống, không đúng thì thẻ không thuộc hệ thống.

## Chứng Thực Thẻ Đối Với Partner (Nếu Có) (Step2)

Chứng thực thẻ của Partner được thực hiện bên Client dựa vào các dữ liệu sau:

* Key Public trong file Config của hệ thống
* Dùng KeyA để đọc License trong thẻ từ sector 4 đến sector 6
* Thư viện PCSC Reader để đọc Serial Number của thẻ

Tiếp theo, sử dụng Key Public để giải mã License theo thuật toán RSA, đối chiếu kết quả giải mã với Serial Number. Nếu đúng thì thẻ thuộc hệ thống, không đúng thì thẻ không thuộc hệ thống.

## Kiểm Tra Tính Hiệu Lực Của Thẻ

/// <summary>

/// Kiểm tra tính hiệu lực của thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="dataPayIn">Dữ liệu nạp tiền sector 11</param>

/// <param name="dataPayOut"> Dữ liệu trừ tiền sector 12</param>

/// <returns> Kết quả trả về gồm:

/// 0: Normal,

/// 1: Locked

/// 2: Canceled,

/// 3: Expired,

/// 4: Broken,

/// 5: Lost

/// 6: LockMoneyCard

/// </returns>

int ValidateCard(string session, string serialNumber, string dataPayIn, string dataPayOut);

### Transaction

### Client (Step3)

* Người dùng đưa thẻ vào đọc, sử dụng thư viện PCSCReader để lấy SerialNumber của thẻ
* Dùng KeyA mặc định để đọc dữ liệu sector 11 (dataPayIn)
* Dùng KeyA mặc định để đọc dữ liệu sector 12 (dataPayOut)
* Gọi hàm **ValidateCard** để lấy dữ liệu

### Server

**Step 4**

* Vào Table **swtgp\_ps\_chip\_personalization** kiểm tra thẻ đã được phát hành hay chưa?
  + Nếu đúng: tiếp tục
  + Nếu sai: Trả về kết quả **Status** (NotPerso) cho Client
* Vào Table **swtgp\_cms\_cardchip** kiểm tra tình trạng của thẻ (Field: **LogicalStatus, PhysicalStatus**) đang ở trạng thái Normal không?
  + Nếu đúng: tiếp tục
  + Nếu sai: Trả về kết quả **Status** (Locked, Canceled, Expired, Broken, Lost) cho Client
* Kiểm tra **Log PayIn yêu cầu nạp tiền** và **Log PayIn yêu cầu nạp tiền** **đã thực hiện thành công** có **trùng khớp** với **dataPayIn** hay không?
  + Nếu không khớp: Cập nhật **Status** (LockMoneyCard) của **Log PayIn yêu cầu nạp tiền** và **Log PayIn yêu cầu nạp tiền đã thực hiện thành công** trước đó. Trả kết quả **Status** (LockMoneyCard) cho Client
  + Nếu trùng khớp: tiếp tục.
* Kiểm tra **Log PayOut yêu cầu trừ tiền** và **Log PayOut yêu cầu trừ tiền đã thực hiện thành công** phải **bé hơn hoặc bằng** **dataPayOut** không?
  + Nếu không khớp: Cập nhật **Status** (LockMoneyCard) của **Log PayIn yêu cầu nạp tiền** và **Log PayIn yêu cầu nạp tiền đã thực hiện thành công** trước đó. Trả kết quả **Status** (LockMoneyCard) cho Client
  + Nếu trùng khớp: tiếp tục.
* Vào Quản lý Key để sinh **KeyA** cho **sector 11, 12**

**Step 5**

* Trả kết quả tình trạng của thẻ cho Client

## Quy Trình Nạp Tiền Vào Thẻ

### Yêu Cầu Thực Hiện Nạp Tiền Vào Thẻ

/// <summary>

/// Yêu cầu thực hiện nạp tiền vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="IpAddress">Địa chỉ Ip của máy trạm giao dịch</param>

/// <param name="amount">Số tiền muốn nạp vào thẻ</param>

/// <param name="code">Mã chứng thực số tiền sector 11</param>

/// <returns> Kết quả trả về PayInDto gồm các field sau:

/// Id

/// KeyB

/// DataWriteToCard

/// VerificationCode

/// Status

/// </returns>

PayInDto GetDataPayInWriteToCard(string session, string serialNumber, string IpAddress, long amount, string code);

#### Transaction

/// <summary>

/// Thực hiện quy trinh nạp tiền gồm 2 giai đoạn:

/// 1. Ghi Log Client yêu cầu nạp tiền

/// 2. Ghi Log Client đã nạp tiền vào thẻ thành công

/// </summary>

public class PayInDto

{

public long Id { get; set; }

/// <summary>

/// 1. Ghi Log Client yêu cầu nạp tiền => filde NULL

/// 2. Ghi Log Client đã nạp tiền vào thẻ thành công sẽ

/// lưu id của Log Client gửi yêu cầu nạp tiền

/// Mục đích: Để mapping Log yêu cầu nạp tiền

/// và Log yếu cầu nạp tiền đã thực hiện thành công

/// </summary>

public long PayInParentId { get; set; }

/// <summary>

/// Khi thực hiện quy trình tất toán mới lưu SettlementId

/// Mục đích để quản lý quy trình tất toán

/// </summary>

public long SettlementId { get; set; }

public long PartnerId { get; set; }

public long MemberId { get; set; }

/// <summary>

/// Địa chỉ IP của máy trạm thực hiện giao dịch

/// </summary>

public string IpAddress { get; set; }

public string SerialNumber { get; set; }

/// <summary>

/// Số tiền được trừ trong thẻ

/// </summary>

public long Amount { get; set; }

public string PayInDate { get; set; }

/// <summary>

/// Username thực hiện yêu cầu trừ tiền

/// </summary>

public string Owner { get; set; }

/// <summary>

/// Dữ liệu được ghi vào sector 11 của thẻ

/// </summary>

public string DataWriteToCard { get; set; }

/// <summary>

/// KeyB của sector 11

/// </summary>

public string KeyB { get; set; }

/// <summary>

/// Mã xác thực của bên thứ 3 (VD: ngân hàng)

/// </summary>

public string VerificationCode { get; set; }

/// <summary>

/// Lưu số lần đồng bộ dữ liệu

/// </summary>

public int SnysDataNumber { get; set; }

/// <summary>

/// 0: Successful

/// 1: Processing

/// 3: LockMoneyCard

/// </summary>

public int Status { get; set; }

}

#### Client

**Step 6**

* Người dùng đưa thẻ vào đọc, sử dụng thư viện **PCSCReader** để lấy **SerialNumber** của thẻ
* Đọc **IpAddress** trong **file config** của hệ thống
* Amount: người nhân viên nhập vào
* Gọi hàm **GetDataPayInWriteToCard** để lấy dữ liệu nạp tiền

**Step 9**

* Dùng **KeyB** để ghi dữ liệu vào **sector 11**

#### Server

**Step 7**

* Lấy dữ liệu trong Table **swtgp\_ps\_chip\_personalization** theo SerialNumber gồm các field: Id, MemberId …
* Lấy dữ liệu trong Table **swtgp\_ps\_member** theo MemberId gồm các field: Id, OrgId …
* Vào quản lý key để sinh **KeyB** cho **sector 11**
* Lưu Log PayIn với các field sau:
  + PartnerId
  + MemberId
  + IpAddress
  + SerialNumber
  + Amount
  + PayInDate
  + Owner (Lấy **username** theo **session**)
  + DataWriteToCard
  + KeyB (**KeyB** của sector 11)
  + Status: Processing

**Step 8**

* Trả kết quả **PayInDto** cho Client gồm các file sau:
  + Id
  + KeyB
  + DataWriteToCard
  + Status

### Cập Nhật Trạng Thái Log Nạp Tiền

/// <summary>

/// Cập nhật trạng thái Log nạp tiền

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="payInId">Id của Log nạp tiền</param>

/// <param name="field">Chưa biết sử dụng làm gi? (Hỏi anh Nam)</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL

/// </returns>

int UpdateStatusPayIn(string session, string payInId, string field);

#### Transaction

#### Client

**Step 10**

* Gọi hàm **UpdateStatusPayIn** với parameter là **payInId** (**payInId** là kết quả trả về của hàm **GetDataPayInWriteToCard**)
* Thông báo yêu cầu đã thực hiện thành công

#### Server

**Step 11**

* Vào table **PayIn** lấy thông tin **Log PayIn yêu cầu nạp tiền** theo điều kiện **payInId** và **Status** (Processing)
* Lưu **Log PayIn yêu cầu đã nạp tiền thành công** gồm các field sau:
  + Các thông tin **Log yêu cầu nạp tiền**
  + PayInParentId (Id của **Log PayIn yêu cầu nạp tiền**)
  + Status (Successfull)

**Step 12**

* Trả kết quả **Status** (SUCCESS) cho Client

## Quy Trình Trừ Tiền Trong Thẻ

### Yêu Cầu Thực Hiện Trừ Tiền Vào Thẻ

/// <summary>

/// Yêu cầu thực hiện trừ tiền vào thẻ

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="memberId">Id của Member</param>

/// <param name="serialNumber">Serial number của thẻ</param>

/// <param name="IpAddress">Địa chỉ Ip của máy trạm giao dịch</param>

/// <param name="amount">Số tiền muốn trừ</param>

/// <param name="code">Mã chứng thực số tiền của sector 11(suy nghĩ thêm về vấn đề này)</param>

/// <returns>Dữ liệu để thực hiện trừ tiền vào thẻ</returns>

PayOutDto GetDataPayOutWriteToCard(string session, string serialNumber, string IpAddress, long amount, string code);

#### Transaction

/// <summary>

/// Thực hiện quy trinh trừ tiền gồm 2 giai đoạn:

/// 1. Ghi Log Client yêu cầu trừ tiền

/// 2. Ghi Log Client đã trừ tiền vào thẻ thành công

/// </summary>

public class PayOutDto

{

public long Id { get; set; }

/// <summary>

/// 1. Ghi Log Client yêu cầu trừ tiền => filde NULL

/// 2. Ghi Log Client đã trừ tiền vào thẻ thành công sẽ

/// lưu Id của Log Client gửi yêu cầu trừ tiền

/// </summary>

public long PayOutParentId { get; set; }

public long PartnerId { get; set; }

public long MemberId { get; set; }

/// <summary>

/// Địa chỉ IP của máy trạm thực hiện giao dịch

/// </summary>

public string IpAddress { get; set; }

public string SerialNumber { get; set; }

/// <summary>

/// Số tiền được trừ trong thẻ

/// </summary>

public long Amount { get; set; }

public string PayOutDate { get; set; }

/// <summary>

/// Username thực hiện yêu cầu trừ tiền

/// </summary>

public string Owner { get; set; }

/// <summary>

/// Dữ liệu được ghi vào sector 12 của thẻ

/// </summary>

public string DataWriteToCard { get; set; }

/// <summary>

/// KeyB của sector 12

/// </summary>

public string KeyB { get; set; }

/// <summary>

/// Mã xác nhận của bên thứ 3 (VD: ngân hàng)

/// </summary>

public string VerificationCode { get; set; }

/// <summary>

/// Lưu số lần đồng bộ dữ liệu

/// </summary>

public int SnysDataNumber { get; set; }

/// <summary>

/// 0: Successful

/// 1: Processing

/// 3: LockMoneyCard

/// </summary>

public int Status { get; set; }

}

#### Client

**Step 6**

* Người dùng đưa thẻ vào đọc, sử dụng thư viện **PCSCReader** để lấy **SerialNumber** của thẻ
* Đọc **IpAddress** trong **file config** của hệ thống
* **Amount** người nhân viên nhập vào
* Gọi hàm **GetDataPayOutWriteToCard** để lấy dữ liệu trừ tiền

**Step 9**

* Dùng **KeyB (Key mặc định)** để ghi dữ liệu vào **sector 12**

#### Server

**Step 7**

* Lấy dữ liệu trong Table **swtgp\_ps\_chip\_personalization** theo SerialNumber gồm các field: Id, MemberId …
* Lấy dữ liệu trong Table **swtgp\_ps\_member** theo MemberId gồm các field: Id, OrgId …
* Vào quản lý key để sinh **KeyB** cho **sector 12**
* Lưu Log PayOut với các field sau:
  + PartnerId (OrgId của table **swtgp\_ps\_member**)
  + MemberId
  + IpAddress
  + SerialNumber
  + Amount
  + PayOutDate
  + Owner (Lấy **username** theo **session**)
  + DataWriteToCard
  + KeyB (**KeyB** của sector 12)
  + Status: Processing

**Step 8**

* Trả kết quả **PayOutDto** cho Client gồm các field sau:
  + Id
  + DataWriteToCard
  + Status

### Cập Nhật Trạng Thái Log Trừ Tiền

/// <summary>

/// Cập nhật trạng thái Log trừ tiền

/// </summary>

/// <param name="session">Mã session của user đã đăng nhập</param>

/// <param name="payInId">Id của Log trừ tiền</param>

/// <param name="field">Chưa biết sử dụng làm gi? (Hỏi anh Nam)</param>

/// <returns>Tình trạng kết quả

/// 0: SUCCESS,

/// 1: FAILED,

/// 2: CANCEL

/// </returns>

int UpdateStatusPayOut(string session, string payOutId, string field);

#### Transaction

#### Client

**Step 10**

* Gọi hàm **UpdateStatusPayOut** với parameter là **payOutId** (**payOutId** là kết quả trả về của hàm **GetDataPayOutWriteToCard**)
* Thông báo yêu cầu đã thực hiện thành công

#### Server

**Step 11**

* Vào table **PayOut** lấy thông tin **Log PayOut** **yêu cầu trừ tiền** theo điều kiện **payOutId** và **Status** (Processing)
* Lưu **Log PayOut** **yêu cầu đã trừ tiền thành công** gồm các field sau:
  + Các thông tin **Log yêu cầu trừ tiền**
  + PayOutParentId (Id của **Log** **PayOut yêu cầu nạp tiền**)
  + Status (Successfull)

**Step 12**

* Trả kết quả **Status** (SUCCESS) cho Client

## Đồng Bộ Dữ Liệu

Ghi chú: Chức năng này sẽ được thảo luận sau