



INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA (ISEL)

DEPARTAMENTO DE ENGENHARIA ELETRÓNICA E DE
TELECOMUNICAÇÕES E COMPUTADORES (DEETC)

LEIM

LICENCIATURA EM ENGENHARIA INFORMÁTICA E MULTIMÉDIA
UNIDADE CURRICULAR DE PROJETO

osTicket-API



António Ferreira 47500

Tomás Gomes 48614

Pedro Silva 48637

Orientador(es)

Professor Doutor Tiago M. Dias

Professor Doutor Carlos Gonçalves

julho, 2024

Resumo

Este projeto, proposto pela **A**utoridade **N**acional de **S**egurança **R**odoviária (ANSR) em colaboração com o **I**nstituto **S**uperior de **E**ngenharia de **L**isboa (ISEL), consistiu em criar e implementar uma ***A**pplication **P**rogramming **I**nterface* (API), instalada através de um *Plugin*, para o *software* de gestão de *tickets*, osTicket, que permita fazer a gestão de *tickets* programaticamente. Durante este desenvolvimento, foi necessário estudar a arquitetura deste *software* e as suas funcionalidades.

Para além de implementar as funcionalidades possíveis através da interface do osTicket, foi necessário criar um novo sistema de autenticação da API e adicionar um novo estado ao sistema, o estado **Suspended**. Foram ainda adicionadas funcionalidades extra para facilitar o uso da API desenvolvida.

O *Plugin* que instala a API não só pode ser utilizado pela ANSR, como também por qualquer pessoa ou organização que o queira integrar no seu *software* de gestão.

Visto que também foi criado e disponibilizado um manual de criação de *Plugins* e um manual de utilização da nossa API, este projeto dá também à comunidade as ferramentas necessárias para que consigam criar e desenvolver as suas próprias versões adaptadas aos seus problemas específicos.

Índice

Resumo	i
Índice	iii
Lista de Figuras	v
Lista de Tabelas	vii
Lista de Siglas e Acrónimos	ix
1 Introdução	1
1.1 Contributos do Trabalho	3
1.2 Organização do Relatório	4
2 Estudo da Arquitetura	5
2.1 Autenticação	5
2.2 <i>Ticket</i>	7
2.3 <i>Plugins</i>	9
3 Modelo Proposto	11
3.1 Requisitos	11
3.2 Abordagem	12
3.3 Nova Autenticação	13
3.4 Suspend <i>Tickets</i>	14
3.5 Funcionalidades extra	16
3.6 Estrutura de um <i>Plugin</i> no osTicket	17
3.7 Instalação de um <i>Plugin</i>	19

4	Implementação do Modelo	27
4.1	Tabelas Extra	27
4.2	Classes	30
4.3	Endpoints	37
4.4	Detalhes do <i>plugin</i>	40
5	Validação e Testes	45
6	Conclusões e Trabalho Futuro	49
A	Criação de um <i>Plugin</i> na Íntegra	51
B	Testes Postman	67
	Bibliografia	113

Lista de Figuras

1.1	Relação entre os módulos do sistema Sistema Nacional de Controlo de Velocidade (SINCRO)	2
2.1	Tabela <code>ost_api_key</code>	6
2.2	Estrutura de um ticket	7
2.3	<i>Thread</i> exemplo de um <i>ticket</i>	8
3.1	Detalhes da entidade <code>api_key_extension</code>	14
3.2	Esquema explicativo da suspensão de <i>Service Level Agreement</i> (SLA)	15
3.3	Detalhes da entidade <code>suspended_ticket</code>	16
3.4	Exemplo de uma <code>checkbox</code>	19
3.5	<i>Login</i> como agente	20
3.6	Selecionar Admin Panel	21
3.7	Selecionar Manage	21
3.8	Selecionar Plugins	22
3.9	Selecionar Add New Plugin	22
3.10	Selecionar o <i>Plugin</i> a instalar	23
3.11	Alterar estado para Active e guardar	23
3.12	Selecionar Add New Instance	24
3.13	Ativar instância	24
3.14	Notificação de sucesso	24
3.15	Desinstalação de um <i>Plugin</i>	25
4.1	Relações da nova tabela <code>api_key_extension</code>	28
4.2	Tabela relativa à nova autenticação.	28
4.3	Relações da nova tabela <code>suspended_extension</code>	29
4.4	Tabela que armazena <i>tickets</i> suspensos.	29

4.5	Tabela com todos os estados do <i>ticket</i>	29
4.6	Tabela com todos os eventos do <i>ticket</i>	30
4.7	Exemplo na interface de <i>ticket</i> suspenso por agente.	30
4.8	Diagrama de classes geral.	31
4.9	Diagrama da classe <code>PluginExtension</code>	32
4.10	Diagrama da classe <code>TableInstaller</code>	32
4.11	Diagrama da Classe <code>PluginConfigExtension</code>	33
4.12	Diagrama Classe <code>ApiExtension</code>	34
4.13	Diagrama da Classe <code>TicketApiControllerExtension</code>	35
4.14	Exemplo de alteração simultânea de vários parâmetros de um <i>ticket</i>	36
4.15	Níveis de prioridades	36
4.16	Opções de origem de <i>tickets</i>	36
4.17	Exemplo Classe <code>PluginExtension</code>	37
4.18	Cabeçalho do pedido com formato <i>JavaScript Object Notation</i> (JSON).	39
4.19	Cabeçalho do pedido para fechar <i>tickets</i> com formato JSON .	39
4.20	Cabeçalho do pedido para para gerar uma nova chave com formato JSON	40
4.21	Selecionar Config	41
4.22	Selecionar Opções de configuração do <i>plugin</i>	42
4.23	Desativar o <i>plugin</i>	43
4.24	Verificar que o <i>plugin</i> está desativado	43
4.25	Apagar o <i>plugin</i>	44
5.1	Pedido de abrir <i>ticket</i> no <i>software</i> Postman	46
5.2	Interface do <code>osTicket</code> com o <i>ticket</i> no estado aberto.	46
5.3	Pedido de suspender <i>ticket</i> no <i>software</i> Postman	46
5.4	Interface do <code>osTicket</code> com o <i>ticket</i> no estado suspenso.	47
5.5	Pedido de fechar <i>ticket</i> no <i>software</i> Postman	47
5.6	Interface do <code>osTicket</code> com o <i>ticket</i> no estado fechado.	47

Lista de Tabelas

3.1	Funcionalidades da API necessárias para a gestão de tickets . . .	12
3.2	Funcionalidades extra	17

Lista de Siglas e Acrónimos

ANSR	Autoridade Nacional de Segurança Rodoviária
API	<i>Application Programming Interface</i>
CSV	<i>Comma-Separated Value</i>
IP	<i>Internet Protocol</i>
ISEL	Instituto Superior de Engenharia de Lisboa
ITSM	Information Technology Service Management
JSON	<i>JavaScript Object Notation</i>
LCV	Local de Controlo de Velocidade
LCVI	Local de Controlo de Velocidade Instantânea
LCVM	Local de Controlo de Velocidade Média
MD5	<i>Message Digest Algorithm 5</i>
REST	<i>Representational State Transfer</i>
SIGET	Sistema de Gestão de Eventos de Trânsito
SIGET-SM	Sistema de Gestão de Eventos de Trânsito Sistema de Manutenção
SIGET-M	Sistema de Gestão de Eventos de Trânsito Monitorização
SINCRO	Sistema Nacional de Controlo de Velocidade

SLA	<i>Service Level Agreement</i>
SOAP	<i>Simple Object Access Protocol</i>
XML	<i>Extensible Markup Language</i>
URL	<i>Uniform Resource Locator</i>
SQL	<i>Structured Query Language</i>

Capítulo 1

Introdução

A ANSR é um serviço central da administração direta do Estado, dotado de autonomia administrativa, que tem por missão o planeamento e coordenação a nível nacional de apoio à política do Governo em matéria de segurança rodoviária, bem como a aplicação do direito contraordenacional rodoviário.

Em colaboração com o ISEL e no âmbito do contrato CE 19/2022 foi proposta uma arquitetura de referência para o novo módulo de gestão integrado no **Sistema de Gestão de Eventos de Trânsito (SIGET)**, da responsabilidade da ANSR, denominado de módulo **Sistema de Gestão de Eventos de Trânsito Sistema de Manutenção (SIGET-SM)**.

O módulo SIGET-SM destina-se a suportar a gestão do serviço de manutenção de todos os **Local de Controlo de Velocidade (LCV)** do SINCRO, sejam eles para os **Local de Controlo de Velocidade Instantânea (LCVI)** ou **Local de Controlo de Velocidade Média (LCVM)**.

Para além do SIGET-SM, existe também o módulo **Sistema de Gestão de Eventos de Trânsito Monitorização (SIGET-M)**, baseado no sistema Zabbix, é responsável por monitorizar os LCV existentes e comunicar problemas encontrados aos prestadores de serviço para estes abrirem um *ticket* sobre o problema encontrado no módulo SIGET-SM.

Assim, o módulo SIGET-SM é utilizado para efetuar a gestão integrada e completa de pedidos de intervenção de manutenção solicitados tanto pela ANSR como pelas entidades responsáveis pela realização da manutenção dos equipamentos.

A interação entre todos os módulos anteriormente referidos está representada na Figura 1.1, onde se pode observar que a ANSR envia e recebe

informações dos dois módulos e dos prestadores de serviços.

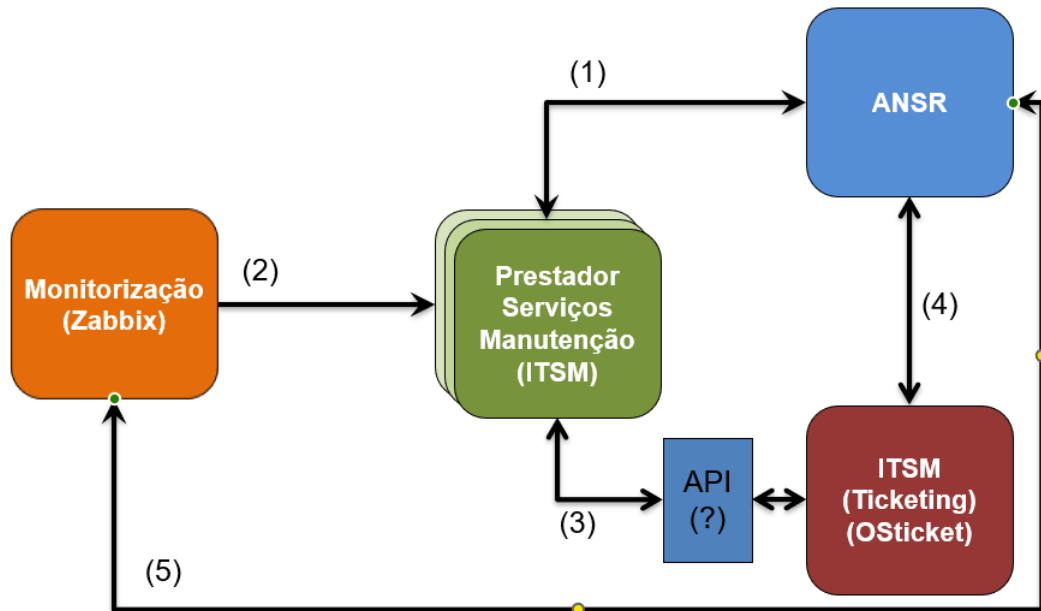


Figura 1.1: Relação entre os módulos do sistema SINCRO: 1) Interação com Prestador (pedidos, incidentes, relatórios, pontos de situação do serviço) 2) Alertas Monitorização; 3) Replicação de Abertura e Fecho de *Tickets*; 4) Relatórios / Vistas On-line dos Serviços; 5) Relatórios / Vistas On-line da Disponibilidade de Rede.

A ANSR como entidade reguladora consulta os relatórios gerados pelos módulos e utiliza as interfaces dos módulos (vistas). A interação entre a ANSR e os prestadores de serviço inclui a consulta de relatórios relativos à execução dos contratos ou ponto de situação dos problemas reportados pelos módulos SIGET-M e SIGET-SM.

Os prestadores de serviços ao receberem o alerta enviado pelo sistema Zabbix, registam o problema no módulo de gestão interno (**I**nformation **T**echnology **S**ervice **M**anagement (ITSM) do prestador de serviço) e de seguida registam o problema no módulo SIGET-SM (osTicket) manualmente.

No contexto do projeto SINCRO foi decidido pela ANSR utilizar o sistema osTicket na sua versão 1.8 para suportar o módulo ITSM.

O osTicket é uma plataforma de *helpdesk* de código aberto que permite a gestão de *tickets* de forma eficiente durante todo o seu ciclo de vida, desde

a sua criação, à sua atribuição, o seu acompanhamento e ao seu fecho, facilitando a comunicação entre as duas entidades.

Dentro das suas funcionalidades, além da gestão de *tickets*, é também importante referir que este *software* fornece métricas e relatórios detalhados tais como desempenho das equipas de suporte ou tempo médio de resposta, entre outros, permitindo uma análise detalhada de todo o serviço.

O osTicket é um sistema flexível, que permite que quem o utilize, o adapte às suas necessidades específicas através de *Plugins*. A possibilidade de utilizar mais funcionalidades do osTicket programaticamente é bastante importante para facilitar a automatização da gestão dos *tickets*, no entanto, o osTicket apenas permite a abertura de *tickets* através da API.

Neste momento a interação entre os vários módulos tem de ser manual, principalmente a interação entre os prestadores de serviço e o SIGET-SM (osTicket), pois os prestadores de serviço têm de fazer a manutenção de cada *ticket* na sua plataforma de gestão privada e no osTicket. A implementação de uma API mais completa (que disponibilize a maioria das funcionalidades que a interface do osTicket permite realizar) iria permitir a automatização de gestão do osTicket diminuindo assim o trabalho dos prestadores de serviço.

O presente trabalho implementa um *plugin* para o osTicket, que estende as funcionalidades existentes de modo a ser possível gerir todo o ciclo de vida de um ticket de forma programática.

1.1 Contributos do Trabalho

Como resultado do trabalho descrito neste relatório obteve-se um *plugin* que irá permitir à ANSR a integração dos sistemas de ITSM dos diferentes prestadores com o módulo SIGET-SM da ANSR (c.f. Figura 1.1). No entanto, o *plugin* é suficientemente genérico para poder ser utilizado em qualquer instância do osTicket (versão 1.8).

Todo o código desenvolvido e a respetiva documentação está disponível em [Grupo15-online_rep, 2024]. O manual de instalação e de utilização disponível em [Grupo15, 2024].

1.2 Organização do Relatório

Além deste capítulo de introdução o relatório está organizado no seguinte modo. No Capítulo 2 é apresentado o estudo da arquitetura do osTicket. No Capítulo 3 apresenta-se o modelo proposto. A implementação do modelo proposto é apresentada no Capítulo 4. O funcionamento da API é apresentado no Capítulo 5. As conclusões e trabalho futuro são apresentados no Capítulo 6.

Capítulo 2

Estudo da Arquitetura

Como foi referido no Capítulo 1, a versão mais recente do *software* do osTicket (1.8) apenas disponibiliza a funcionalidade de abrir *tickets* via API. É igualmente importante frisar que atualmente a documentação relacionada com o osTicket é muito escassa e incompleta e neste âmbito, de modo a ser possível implementar todas as funcionalidades necessárias para cumprir os requisitos, foi necessário realizar um estudo da arquitetura deste software.

No início do estudo, foram analisadas todas as funcionalidades disponíveis a partir da interface do osTicket relacionadas com as operações possíveis de realizar sobre *tickets*, sendo elas **abrir**, **fechar**, **editar**, **reabrir** e **eliminar**.

De seguida, foi analisada a API já existente no osTicket, incluindo o seu o protocolo de autenticação e mecanismo de abertura de *tickets*, que serviu de base para o resto das funcionalidades desenvolvidas, como se exemplifica na Secção 2.1. A estrutura de um *ticket* e no que este consiste, mostrado na Secção 2.2. E por fim, foi estudado em que consiste um *plugin*, no contexto do osTicket, apresentado na Secção 2.3.

2.1 Autenticação

O protocolo de autenticação da API disponibilizado pelo osTicket associa uma chave gerada a um endereço de *Internet Protocol* (IP) de modo a que apenas pedidos vindos desse IP apenas tenham sucesso caso a chave presente na base de dados corresponda.

A chave é gerada através da concatenação do tempo atual, com o endereço IP definido, com uma string aleatória de 16 caracteres, antes de ser feita a

concatenação, é aplicada à string a função *Message Digest Algorithm 5* (MD5), que gera uma string que representa um número hexadecimal de 32 caracteres. Depois da concatenação ser feita é aplicada outra vez a função MD5 e depois uma função que transforma todos caracteres em maiúsculas, este processamento está ilustrado na Listagem 1.

Listagem 1: Função php que gera a chave.

```

1 <?php
2 class Misc {
3     public static function randCode($length) {
4         $characters =
5             '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
6         $charactersLength = strlen($characters);
7         $randomString = '';
8         for ($i = 0; $i < $length; $i++) {
9             $randomString .= $characters[rand(0, $charactersLength - 1)];
10        }
11        return $randomString;
12    }
13 }
14 $vars = array('ipaddr' => '192.168.1.1');
15
16 $result = strtoupper(md5(time() . $vars['ipaddr'] .
17     md5(Misc::randCode(16))));
18 echo $result;
19 ?>

```

Esta chave é então colocada no *header* do pedido juntamente com o tipo de formato a ser usado (JSON, *Extensible Markup Language* (XML) ou email).

Como pode ser observado na Figura 2.1, é possível ver a tabela (*ost_api_key*) com as suas respectivas colunas, incluindo um exemplo onde a chave está ativa e associada a um endereço de IP com permissões para criar *tickets* (coluna *can_create_tickets*).

id	isactive	ipaddr	apikey	can_create_tickets	can_exec_cron	notes	updated	created
1	1	127.0.0.1	C7B1E9238965F5D18F69A1019CD661F7	1	0		2024-05-30 20:43:39	2024-05-30 20:43:39

Figura 2.1: Tabela *ost_api_key*

Este mecanismo é limitado, no sentido em que ter uma chave associada a um IP limita um agente a fazer a manutenção de *tickets* a um dispositivo, ou rede, que corresponda ao IP da chave, por este motivo, no *plugin* que

foi desenvolvido, é proposta uma nova solução que associa uma chave a um agente, esta solução vai ser explicada com mais detalhe na Secção 3.3.

Como já tinha sido referido anteriormente, a API atual só permite criar *tickets*, e esta funcionalidade necessita desta autenticação. Para criar novos *tickets*, existem campos obrigatórios a serem preenchidos, sendo eles:

- user**, nome do utilizador;
- email**, email do utilizador;
- help topic**, topico do *ticket*;
- issue summary**, descrição do *ticket* em texto;

Os restantes campos são opcionais, se não forem preenchidos são calculados ou associados com valores definidos por omissão.

2.2 *Ticket*

A Figura 2.2 apresenta a estrutura de um ticket, onde pode ser observado que um *ticket* é formado por dois campos, o cabeçalho e o corpo (*thread*).



Status:	Open	User:	 antFerreira (1)  (Manage Collaborators)
Priority:	Normal	Email:	teste@teste.com
Department:	Support	Source:	API
Create Date:	6/2/24 7:48 PM		
Assigned To:	— Unassigned —	Help Topic:	General Inquiry
SLA Plan:	Default SLA	Last Message:	6/2/24 7:48 PM
Due Date:	6/5/24 8:00 AM	Last Response:	

Figura 2.2: Estrutura de um ticket

Da análise da Figura 2.2, pode-se ver que um *ticket* tem os seguintes elementos:

- status**, indica o estado do *ticket*, por exemplo, se está **aberto** (*open*) ou **fechado** (*closed*);

- priority**, existindo 4 níveis de prioridades diferentes, por omissão, sendo elas **baixo** (*low*), **normal**, **elevado** (*high*) e **emergência** (*emergency*);

- department**, corresponde ao departamento a que o *ticket* está associado;

- create date**, data em que o *ticket* foi criado;

- user**, nome do utilizador que criou o *ticket*;

- email**, email do utilizador que criou o *ticket*;

organization, corresponde à organização a que o *ticket* está associado;
source, indica em que meio *ticket* foi criado, pode ser através da API, interface ou até mesmo por telemóvel;

assigned to, campo que indica a pessoa que está responsável por resolver o *ticket*;

sla plan, *service level agreement*, tempo limite de resolução do *ticket*;

due date, data máxima de resolução do *ticket* calculada com base no tempo de criação do *ticket* e do seu respectivo *sla plan*;

help topic, permite filtrar as informações ao enviar para o suporte indicado, por omissão o osTicket faz a distinção entre quatro tópicos, sendo eles, **feedback**, **general inquiry**, **report a problem** e **report a problem / access issue**;

last message, no campo *thread* mostra as mensagens que foram feitas dentro de um *ticket* e em caso de um utilizador escrever uma mensagem vai ser registado a data e hora da última mensagem neste elemento;

last response, mostra a data e hora da última resposta;

Já no campo *thread*, são mostradas as mensagens de *feedback* relacionadas com o *ticket*, como podemos observar na Figura 2.3.

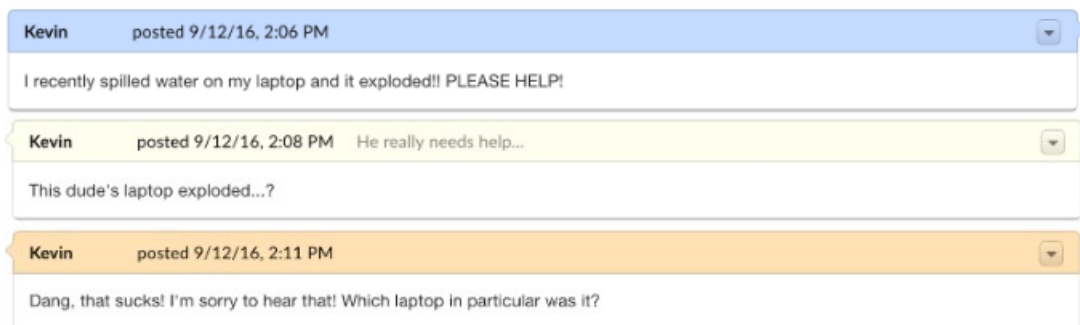


Figura 2.3: *Thread* exemplo de um *ticket*

Cada mensagem é distinguida através de um código de cores: i) azul, dono do *ticket*; ii) amarelo, comentários internos ou ações feitas ao *ticket* (como por exemplo mudar o estado); iii) laranja, respostas dadas pelo agente ao dono do *ticket*;

É importante referir que um *ticket* apenas permite a alteração dos elementos do cabeçalho.

2.3 *Plugins*

O módulo osTicket tem a capacidade de adicionar funcionalidades extras ao sistema sem afetar o código original, tirando partido do conceito de *plugin*. Um *plugin* é constituído por três ficheiros, sendo eles cabeçalho, configuração e execução.

No cabeçalho é mencionado o título, autor, versão, identificação, descrição e referencia para a classe do ficheiro de execução.

A configuração é opcional, podendo ser vazia. Nas situações em que são necessárias configurações, as mesmas podem ser recolhidas através de formulários e os valores recolhidos podem ser utilizados mais tarde.

A lógica de execução do *plugin* é efetuada na classe que suporta o *plugin*, mais concretamente no método **Bootstrap**.

No próximo capítulo serão apresentados os detalhes de implementação de um *plugin*.

Capítulo 3

Modelo Proposto

Neste capítulo vai ser explicado o processo de desenvolvimento da API, começando pela descrição dos Requisitos Funcionais na Secção 3.1, seguido pela abordagem utilizada, na Secção 3.2.

O novo mecanismo de autenticação é explicado na Secção 3.3. O novo estado suspender *tickets* é explicado na Secção 3.4. A estrutura de um *plugin* e os passos necessário para instalar um *plugin* são apresentados respetivamente nas Secções 3.6 e 3.7.

3.1 Requisitos

O objetivo deste trabalho é implementar um modo de efetuar a gestão de *tickets* do osTicket programaticamente, utilizando uma API, de forma a evitar que os prestadores de serviço tenham de repetir o trabalho manual de gerir *tickets* em plataformas diferentes.

Para tal, existem funcionalidades que são necessárias implementar para que seja possível ser feita a gestão de *tickets* através da API. As funcionalidades necessárias, que neste momento só são possíveis fazer manualmente através da interface do osTicket são as seguintes, **abrir**, **fechar**, **editar** e **reabrir** *tickets*. Outra funcionalidade que também foi necessária implementar é a funcionalidade **suspender** *tickets*, que não existe no osTicket. Na Tabela 3.1 são apresentados os requisitos mínimos que possibilitam a gestão de *tickets* através da API.

Tabela 3.1: Funcionalidades da API necessárias para a gestão de tickets

Operação	Funcionalidade	Descrição
Abrir <i>Ticket</i>	Abre novos <i>tickets</i>	Criar <i>tickets</i> utilizando o novo mecanismo de autenticação (Secção 3.3).
Fechar <i>Ticket</i>	Fecha <i>tickets</i>	Fechar um <i>ticket</i> quando o problema no radar respetivo ao <i>ticket</i> tiver sido resolvido
Editar <i>Ticket</i>	Edita <i>tickets</i>	Alterar valores de campos de um <i>ticket</i> (Secção 2.2)
Reabrir <i>Ticket</i>	Reabre <i>tickets</i>	Reabrir <i>tickets</i> que tenham sido fechados e não tenham sido resolvidos, que normalmente acontece por engano do agente.
Suspender <i>Ticket</i>	Suspende <i>tickets</i>	Suspender <i>ticket</i> em caso de falta de recursos necessários para o resolver.

Existem ainda mais funcionalidades que o osTicket permite fazer manualmente através da sua interface gráfica, como consultar departamentos e apagar *tickets*. As funcionalidades extra implementadas podem ser observadas na Secção 3.5.

3.2 Abordagem

Para construir uma API, é preciso ter em conta diversos fatores tais como o protocolo utilizado pela API para fazer a comunicação entre serviços web e a estrutura das mensagens que a API utiliza.

A escolha do protocolo e da estrutura das mensagens, foi principalmente baseada na forma como a API original do osTicket estava desenvolvida, de forma a manter coerência e compatibilidade entre as diferentes API.

Em relação ao protocolo escolhido, foi utilizado o ***Representational State Transfer*** (REST). Este protocolo, para além de ser já utilizado originalmente pelo osTicket, é também mais leve e escalável que a opção ***Simple Object Access Protocol*** (SOAP).

Em relação à estrutura das mensagens, os pedidos feitos por um cliente necessitam de um cabeçalho e o corpo da mensagem. No cabeçalho é indicado

a formatação da informação do corpo da mensagem (JSON) e a chave do agente que contém as permissões necessárias para fazer o respetivo pedido. No corpo é enviado a informação necessária para realizar o pedido.

O corpo das mensagens dos pedidos feitos para o servidor podem ter os formatos JSON, XML e email, sendo que as respostas do servidor para o cliente são enviadas como uma *string*, com toda a informação sobre se o pedido for bem sucedido ou não.

3.3 Nova Autenticação

Na Secção 2.1 foi referido que o osTicket disponibiliza uma autenticação em que é associada uma chave a um endereço de IP. Assim, com a utilização deste mecanismo, levanta-se a seguinte questão, “e se o agente que utiliza a API tiver um endereço de IP diferente ao que está associado à chave?”.

Surge assim a nossa solução de uma nova autenticação em que, em vez de ser associada uma chave a um endereço de IP, esta é associada a um agente.

É importante também referir que também foi necessário adicionar mais atributos à chave de modo a que fosse possível realizar novas operações sobre os *tickets*, não suportadas anteriormente, sendo estas as operações **fechar**, **editar**, **reabrir** e **suspender** *tickets*.

Na Figura 3.1, pode ser observada a entidade `api_key`, que representa uma nova chave, e os seus respetivos atributos, que representam, a própria chave (`apikey`), o identificador do agente à qual a chave pertence (`id_staff`), todas as suas permissões relativas à gestão de *tickets* (`can_create_tickets`, `can_close_tickets`, `can_reopen_tickets`, `can_edit_tickets` e `can_suspend_tickets`), notas adicionais (`notes`), última data de atualização (`updated`), data de criação da chave (`created`), identificador da chave (`id`) e se a chave está ativa ou não (`isactive`).

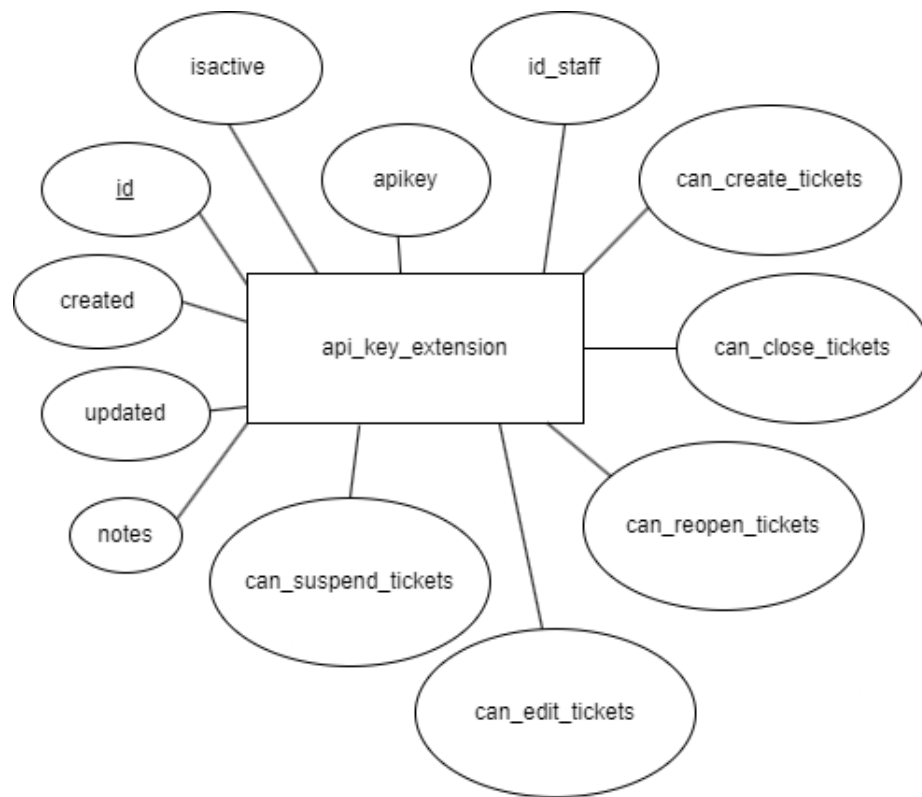


Figura 3.1: Detalhes da entidade `api_key_extension`

O funcionamento da nova autenticação segue o mesmo princípio da antiga em que, a chave é passada no *header* do pedido no formato JSON.

Caso o agente queira uma nova chave por motivos de segurança, após fazer o pedido e ser gerada uma nova chave, a chave antiga é desativada fazendo com que um agente apenas tenha uma e uma só chave ativa.

Na Secção 4.1 são explicadas em mais detalhe todas as alterações feitas à base de dados de forma a permitir o correto funcionamento desta nova autenticação. É também explicado nas Secções 4.3 e 4.4 formas de gerar esta chave.

3.4 Suspende *Tickets*

Antes de falar da suspensão de *tickets* é necessário introduzir o conceito de SLA. Um SLA é um contrato formal entre um fornecedor de serviços e um cliente que define o nível de serviço esperado do fornecedor de serviços. Este

acordo descreve as métricas pelas quais o serviço é medido, as responsabilidades de cada parte, e as soluções ou penalidades em caso de, os níveis de serviço acordados não serem cumpridos.

Neste contexto, o SLA é uma data estimada baseada num acordo entre os prestadores de serviço e a ANSR, sobre o tempo necessário para resolver os problemas nos radares.

Sendo assim, a funcionalidade de suspender *tickets* é utilizada para suspender a contagem do tempo do *ticket* calculado através do seu SLA, fazendo com que a data estimada para a resolução de um *ticket* seja atualizada quando é removida a suspensão do *ticket*.

A Figura 3.2 representa um exemplo da suspensão de um *ticket* com um SLA de 12 horas. Na marca das 03:00 o *ticket* é suspenso e às 07:00 é removida a suspensão, criando assim um intervalo de 4 horas em que o tempo não é contado. Este tempo não contado é então acrescentado à data de conclusão do *ticket*, estendendo-a.

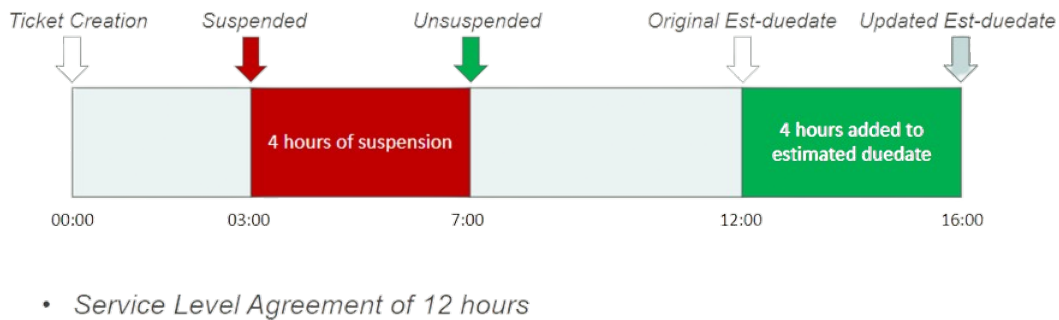


Figura 3.2: Esquema explicativo da suspensão de SLA

No contexto do projeto SINCRO, este conceito, de suspensão de *tickets*, é utilizado nas situações em que um prestador de serviço não consegue avançar com a resolução de um *ticket* (associado a um problema num determinado radar) por razões fora do seu controlo, como estar à espera de autorização para ir à estrada onde o radar se encontra, ou por falta de peças ou equipamentos para poder resolver o problema.

Na Figura 3.3 é apresentada a entidade `suspended_ticket`, e os seus atributos, sendo estes, o número do *ticket* suspenso (`number_ticket`), a data em que o *ticket* foi suspenso (`date_of_suspension`) e a data em que o *ticket* voltou a estar aberto (`date_end_suspension`).

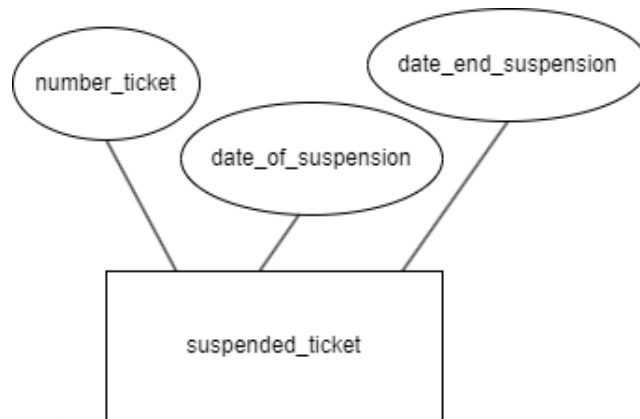


Figura 3.3: Detalhes da entidade `suspended_ticket`

Na Secção 4.1 é explicada a implementação desta tabela na base de dados bem como linhas adicionais noutras tabelas de modo a que seja possível utilizar o novo estado **Suspend**.

3.5 Funcionalidades extra

Para além dos requisitos mínimos, que permitem fazer a gestão de *tickets*, foram ainda implementadas outras funcionalidades oferecidas pelo osTicket, de forma a expandir ao máximo a utilização do osTicket programaticamente, assim como novas funcionalidades, não existentes no osTicket, que facilitem o uso da API desenvolvida. Estas funcionalidades podem ser observadas na Tabela 3.2

Tabela 3.2: Funcionalidades extra

Operação	Funcionalidade
Gerar API <i>Key</i>	Gera uma nova API <i>key</i> para um agente.
Obter API <i>Key</i>	Obtém a chave associada a um agente.
Apagar <i>Ticket</i>	Apaga <i>tickets</i> .
Obter Departamentos	Obtém uma lista com os <i>ids</i> dos departamentos disponíveis e outras informações relevantes.
Obter SLAs	Obtém uma lista com os <i>ids</i> dos SLAs disponíveis e outras informações relevantes.
Obter Agentes	Obtém uma lista com os <i>ids</i> dos agentes disponíveis e outras informações relevantes.
Obter Utilizadores	Obtém uma lista com os <i>ids</i> dos utilizadores disponíveis e outras informações relevantes.
Obter Prioridades	Obtém uma lista com os <i>ids</i> das prioridades disponíveis e outras informações relevantes.
Obter Tópicos	Obtém uma lista com os <i>ids</i> dos tópicos disponíveis e outras informações relevantes.
Obter <i>Sources</i>	Obtém lista com as <i>sources</i> disponíveis.
Obter <i>Tickets</i>	Obtém lista com os números de identificação dos <i>tickets</i> disponíveis e outras informações relevantes.

3.6 Estrutura de um *Plugin* no osTicket

Como introduzido na Secção 2.3, o osTicket permite a instalação de *plugins* que têm como objetivo estender as suas funcionalidades.

É então com base neste conceito que é necessário saber como criar um

plugin. Um *plugin* capaz de ser utilizado pelo osTicket precisa de seguir um conjunto de regras para que este seja reconhecido e utilizável pelo *software*.

Em primeiro lugar, para que o osTicket reconheça um *plugin*, este tem de ter obrigatoriamente dois ficheiros, `plugin.php` e `config.php`. Como se pode observar na Listagem 2, o ficheiro `plugin.php` contém uma descrição geral do *plugin*.

Listagem 2: Código plugin.php

```

1
2 <?php
3 return array(
4     'id' => 'proposal-15:extension',
5     'version' => '0.1',
6     'name' => 'OSTicket API Extension',
7     'author' => 'Grupo Proposta 15',
8     'description' => 'Adds extra functionality to OSTicket API.',
9     'plugin' => 'extension.php:PluginExtension'
10 );
11 ?>

```

No ficheiro `config.php`, apresentado na Listagem 3, que estende da classe `PluginConfig`, é necessário existir um método `getOptions()`, e é neste método que é indicado o formulário de configurações de um *plugin*, se for necessário. Este formulário pode conter **break fields**, **text fields**, **checkboxes**, entre outros, que representam o tipo de de informação apresentada no formulário com as configurações.

É também importante notar que um *plugin* necessita de ter uma instância para poder ser configurado, pois é através de uma instância de um *plugin*, que pode ser feita ou não feita a configuração de dados para garantir o seu correto funcionamento. O mesmo *plugin* pode ter várias instâncias configuradas de forma diferente.

Listagem 3: Ficheiro config.php

```

1
2 <?php
3 class PluginConfigExtension extends PluginConfig {
4     function getOptions() {
5         return array(
6             'save_info' => new BooleanField(array(
7                 'id' => 'save_info',
8                 'label' => __('Save New Tables Info'),
9                 'default' => false,
10                'configuration' => array(
11                    'desc' => __('Saves all values inside the tables added
12                        by this plugin after deactivating it,
13                        so when the plugin is activated again it has all the
14                        same data as before.')
15                )
16            )
17        ),
18    }
19 }

```

```
15         );  
16     }  
17 }  
18 ?>
```

Na Figura 3.4 podemos observar a **checkbox** referida na Listagem 3.

Save New Tables Info:

☐ Saves all values inside the tables added by this plugin after deactivating it, so when the plugin is activated again it has all the same data as before.

Figura 3.4: Exemplo de uma **checkbox**.

Com a existência dos ficheiros referidos anteriormente é então possível criar um *plugin*. No entanto, além da criação do *plugin*, falta ainda o ficheiro que contém a lógica do *plugin*. Neste exemplo, o ficheiro denomina-se **extension.php**. Este ficheiro contém então a lógica do *plugin* desenvolvido e é necessário que seja introduzido no **array** demonstrado na Listagem 3, na opção “plugin”. Na Listagem 4 pode ser observado um exemplo simples da classe do novo *plugin* criado com a função **bootstrap**. Esta função serve para inicializar informações para que o *plugin* possa usar, como por exemplo, obter dados de um ficheiro de configuração. Sempre que o *plugin* está ativo e é chamado, a função **bootstrap** é executada.

Listagem 4: Código extension.php

```
1  
2 <?php  
3 class PluginExtension extends Plugin  
4 {  
5     var $config_class = 'PluginConfigExtension';  
6  
7     function bootstrap()  
8     {  
9         echo "Hello world!";  
10    }  
11 }  
12 ?>
```

Na Secção 3.7 é apresentada a instalação de um *plugin* servindo de exemplo genérico.

3.7 Instalação de um *Plugin*

As imagens apresentadas nesta secção correspondem à instalação e desinstalação do *plugin* que foi desenvolvido, servindo de exemplo de como instalar

ou desinstalar qualquer outro *plugin*. Para fazer a sua instalação é necessário colocar os ficheiros que constituem o *plugin* na diretoria `include/plugins` da instalação do osTicket.

De seguida, é necessário abrir o *website* do osTicket e fazer *login* como agente, como pode ser observado na Figura 3.5.

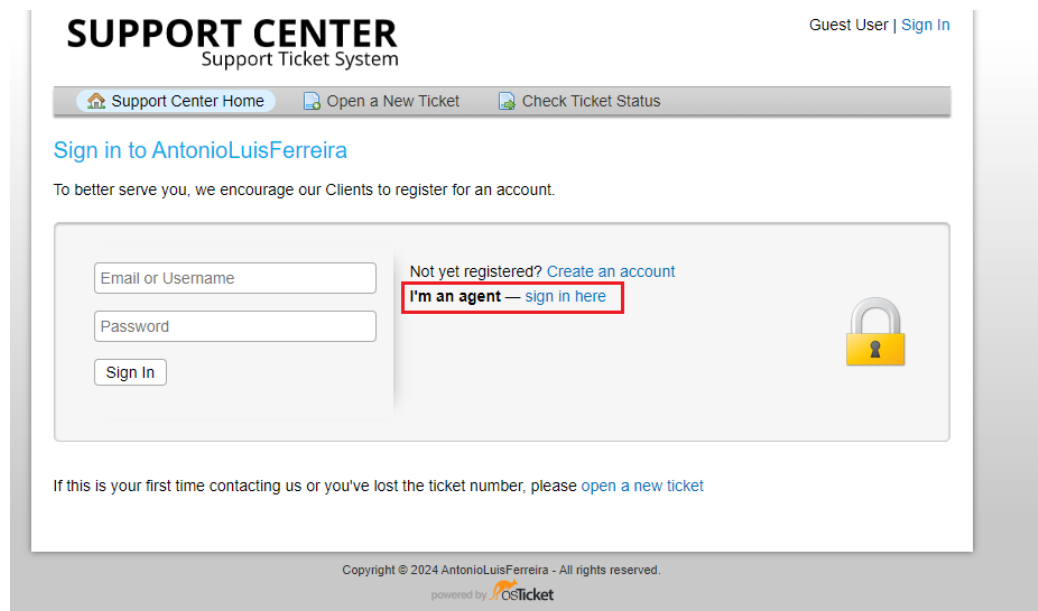


Figura 3.5: *Login* como agente

Já dentro da aplicação, é preciso seleccionar a opção `Admin Panel`, como pode ser observado na Figura 3.6.

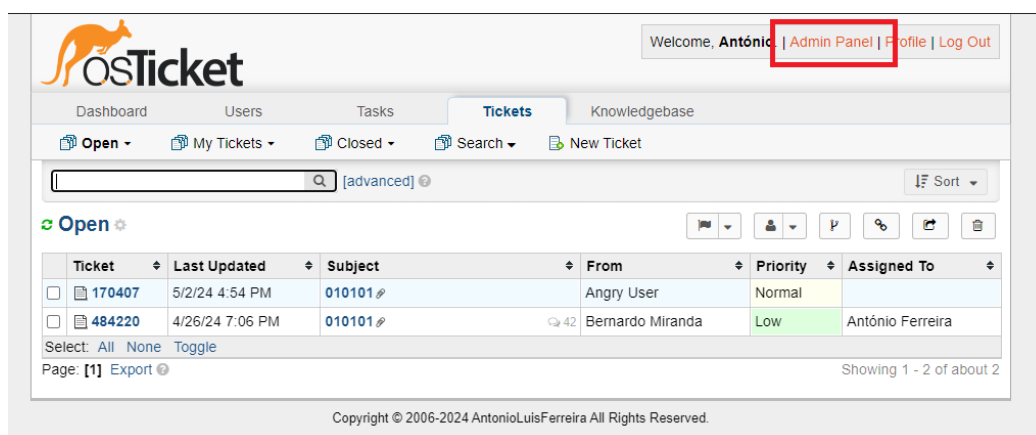


Figura 3.6: Selecionar Admin Panel

De seguida, como podemos observar na Figura 3.7 é necessário seleccionar a aba Manage.

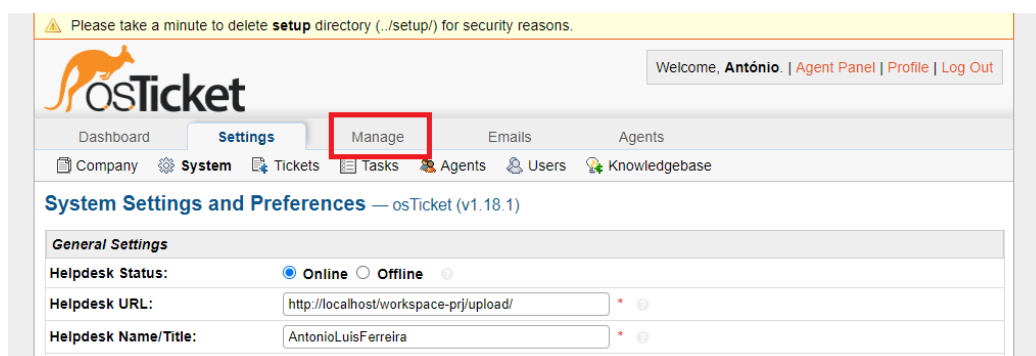


Figura 3.7: Selecionar Manage

É então necessário escolher a opção Plugins, como pode ser observado na Figura 3.8, chegando assim à página Installed Plugins.

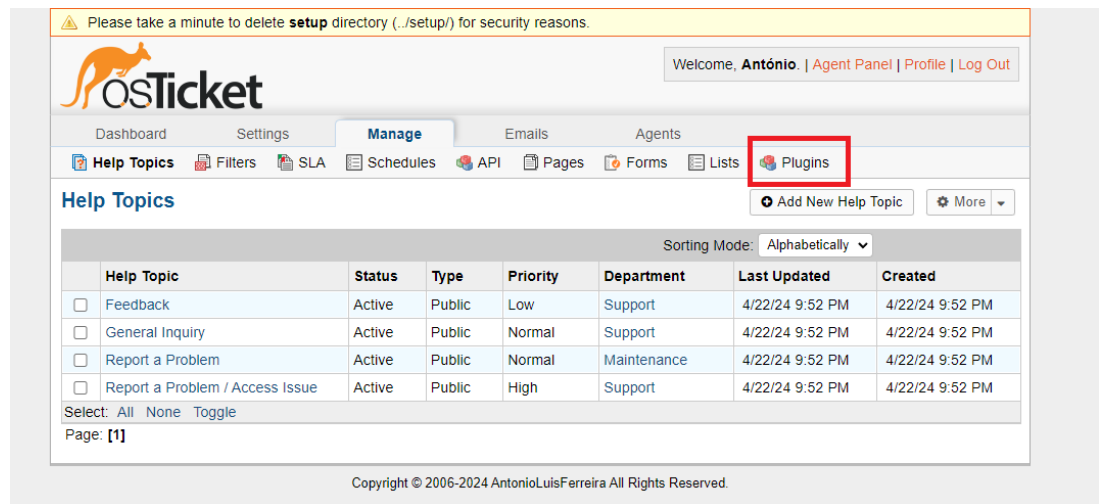


Figura 3.8: Selecionar Plugins

Estando na página de **Installed Plugins**, é preciso selecionar **Add New Plugin**, como pode ser observado na Figura 3.9.

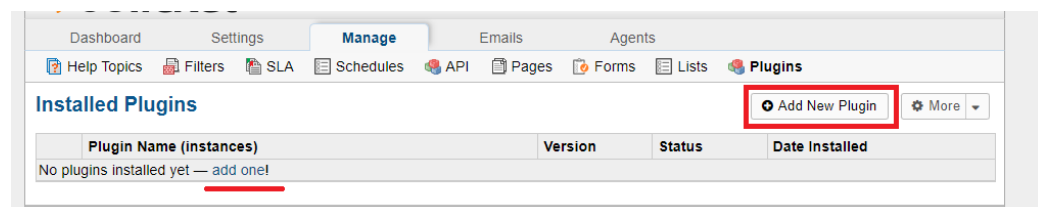
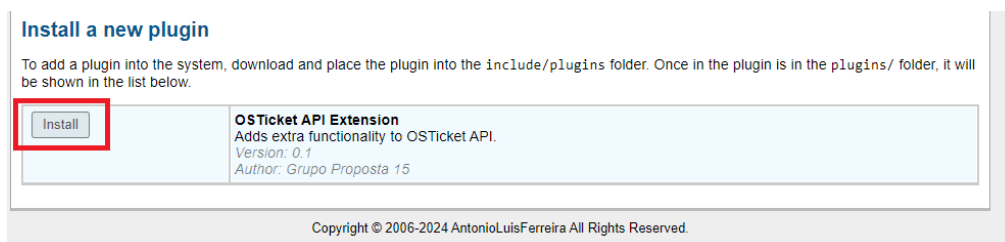
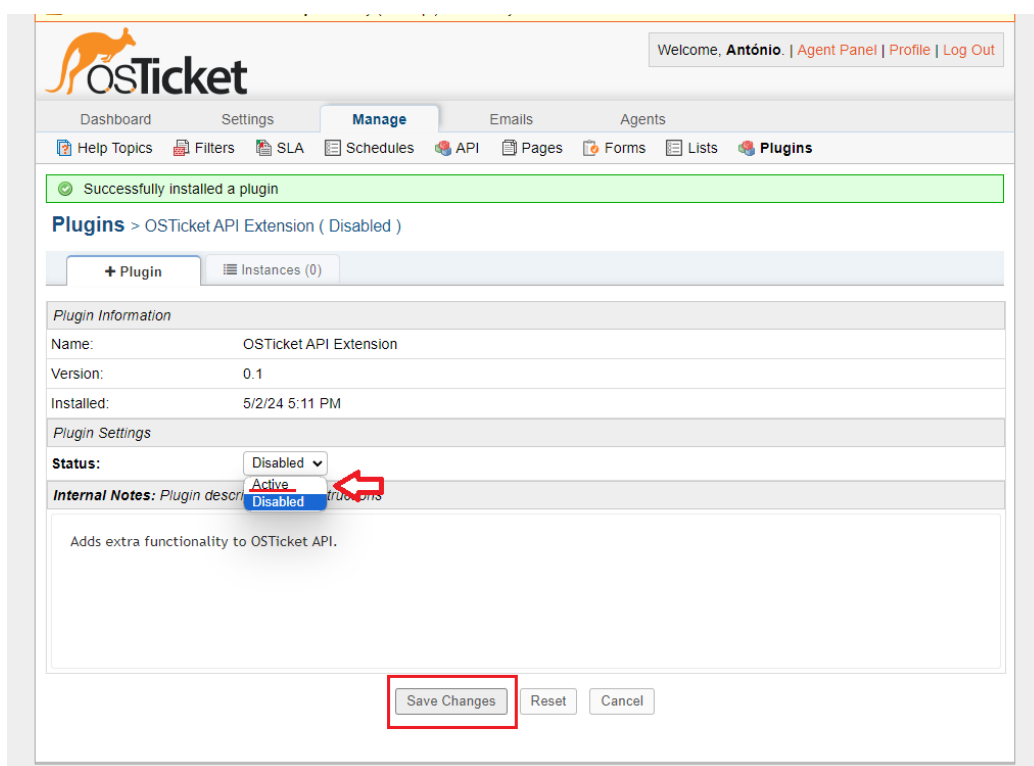


Figura 3.9: Selecionar Add New Plugin

Na página atual, serão mostrados todos os *plugins* disponíveis na diretoria `include/plugins` e que ainda não foram instalados. É necessário então carregar no botão **Install** correspondente ao *plugin* que se deseja instalar, neste caso esse *plugin* é o *plugin* **OSTicket API Extension**, como pode ser observado na Figura 3.10.

Figura 3.10: Selecionar o *Plugin* a instalar

Nas opções de instalação do *plugin*, é preciso selecionar a opção **Status** como **Active** e guardar as alterações, como se pode observar na Figura 3.11.

Figura 3.11: Alterar estado para **Active** e guardar

De seguida, como se pode observar na Figura 3.12, é preciso adicionar uma nova instância carregando no botão **Add New Instance**.

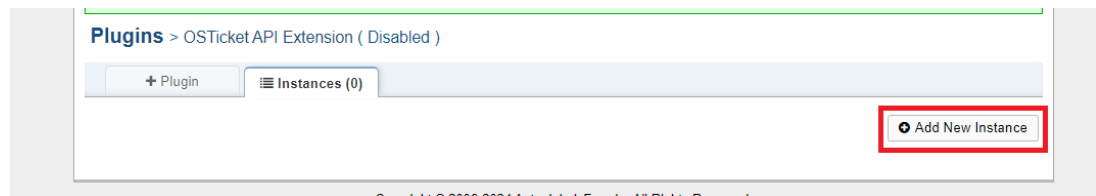


Figura 3.12: Selecionar Add New Instance

É preciso depois escolher um nome que se acha adequado à instância do *plugin*, alterar o seu **Status** para **Enabled**, como se pode observar na Figura 3.13.

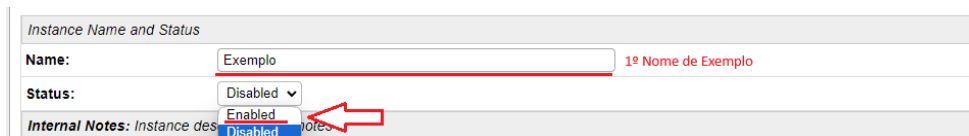


Figura 3.13: Ativar instância

E por fim carregar no botão **Add Instance**. Se o *plugin* tiver sido instalado corretamente, uma notificação de sucesso poderá ser observada no *website* do osTicket, como se pode observar na Figura 3.14.

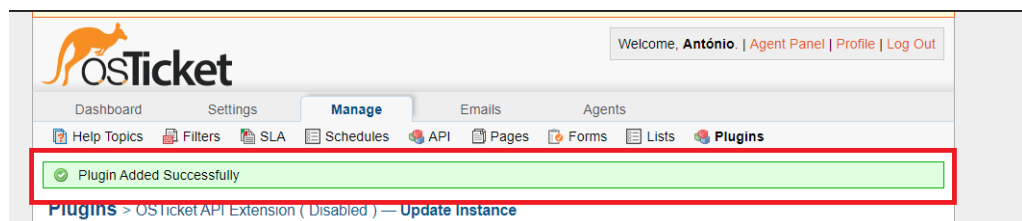
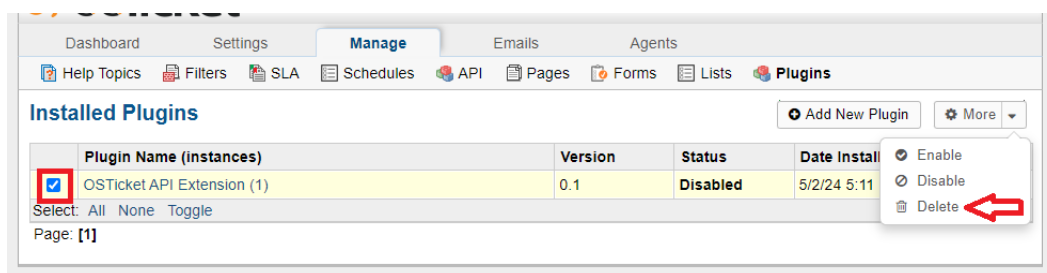


Figura 3.14: Notificação de sucesso

Para desinstalar um *plugin* basta selecionar o *plugin* que se quer desinstalar e selecionar a opção **delete**, como pode ser observado na Figura 3.15.

Figura 3.15: Desinstalação de um *Plugin*

Capítulo 4

Implementação do Modelo

A implementação foi efetuada em paralelo por todos os membros do grupo, com recurso a um Sistema de Controlo de Versões, nomeadamente o `Git` (com o auxílio do `Github`).

Neste capítulo vão ser abordados temas específicos à implementação do *plugin* API, como tabelas extras ao modelo `osTicket` na Secção 4.1, classes desenvolvidas na Secção 4.2, a declaração dos `endpoints` na Secção 4.3 e por fim detalhes específicos ao *plugin* na Secção 4.4.

4.1 Tabelas Extra

Como foi referido na Secção 3.3, foi necessário fazer alterações à base de dados de modo a que esta suporte as novas funcionalidades adicionadas pela instalação do *plugin* desenvolvido, mais particularmente a criação de novas tabelas e inserção de novos valores em tabelas já existentes.

De modo a suportar o novo mecanismo de autenticação, foi criada a relação apresentada na Figura 4.1. Esta nova relação representa que um agente pode ter várias chaves, sendo que só uma pode estar ativa. A entidade `api_key_extension` representa a chave do agente e está encarregue de armazenar o valor da própria chave, o `id` do agente à qual lhe pertence, todas as suas permissões relativas à gestão de *tickets*, notas adicionais, última data de atualização, data de criação e se está ativa ou não.

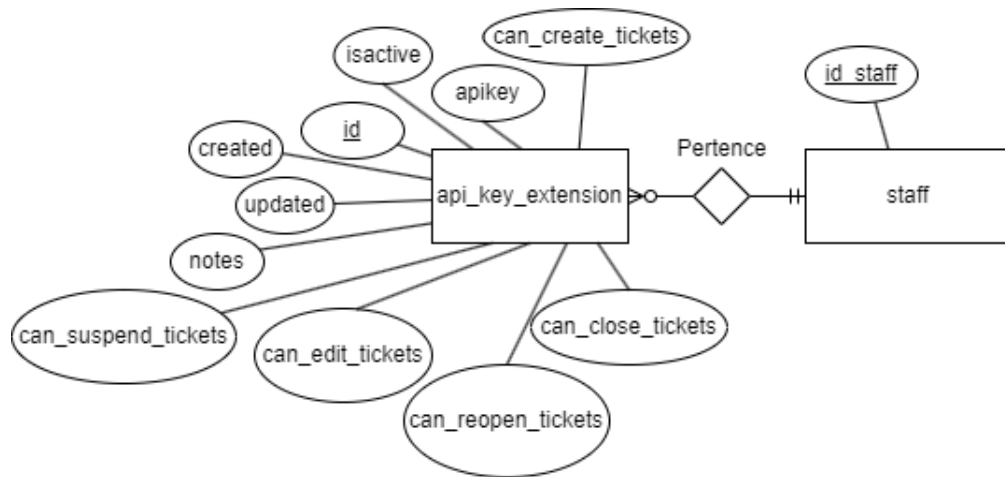


Figura 4.1: Relações da nova tabela `api_key_extension`

Esta relação deu origem à tabela `api_key_extension`, Figura 4.2, com a estrutura apresentada na Secção 3.3.

```
SELECT * FROM `ost_api_key_extension`
```

id	isactive	id_staff	apikey	can_create_tickets	can_close_tickets	can_reopen_tickets	can_edit_tickets	can_suspend_tickets
1	1	1	7EB34A20ADEF89586945DB91853D4219	1	1	1	1	1

Figura 4.2: Tabela relativa à nova autenticação.

Foi também necessário criar a relação `suspended_tickets`, Figura 4.3 para armazenar informações relacionadas aos *tickets* suspensos, como foi referido na Secção 3.4. Como se pode observar na Figura 4.4, esta relação representa a mudança de estado de um *ticket*, e resulta numa nova entidade associativa, que armazena o número do *ticket*, a data em que o *ticket* foi suspenso e a data em que a suspensão foi removida. É importante referir que, o mesmo *ticket* pode ser suspenso quantas vezes forem necessárias.

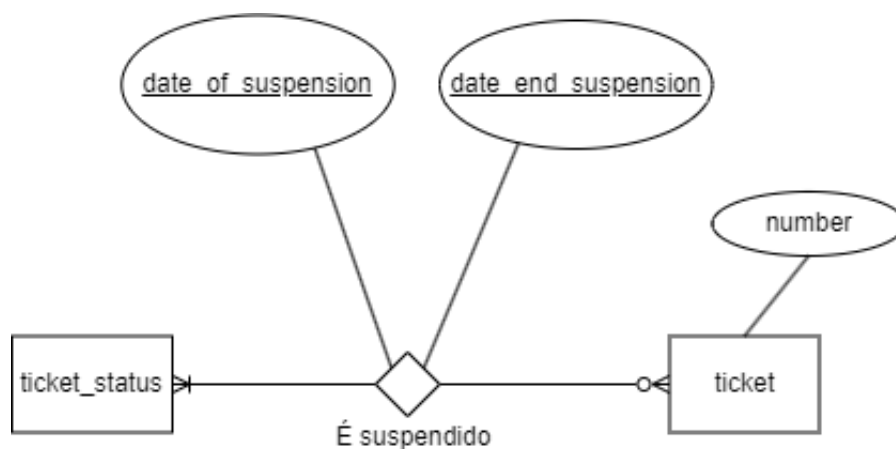


Figura 4.3: Relações da nova tabela `suspended_extension`

Esta relação deu origem à tabela `suspended_ticket`, Figura 4.4, com a estrutura apresentada na Secção 3.4.

```
SELECT * FROM `ost_suspended_ticket`
```

number_ticket	date_of_suspension	date_end_suspension
536791	2024-07-04 18:09:40	2024-07-04 18:10:30

Figura 4.4: Tabela que armazena *tickets* suspensos.

Em relação a tabelas já existentes, foi alterada a tabela `ticket_status` que define os estados do *ticket* (`Open`, `Closed`, `Resolved`, `Archived` e `Deleted`) de modo a suportar o novo estado (`Suspended`), como se pode observar na Figura 4.5.

```
SELECT * FROM `ost_ticket_status` ORDER BY `id` DESC
```

id	name	state	mode	flags	sort	properties	created	updated
6	Suspended	open	3	0	6	{"description":"Tickets are still open but time is..."}	2024-05-21 22:29:52	2024-05-21 22:29:52
5	Deleted	deleted	3	0	5	{"description":"Tickets queued for deletion. Not a..."}	2024-03-15 15:55:51	0000-00-00 00:00:00
4	Archived	archived	3	0	4	{"description":"Tickets only administratively avail..."}	2024-03-15 15:55:51	0000-00-00 00:00:00
3	Closed	closed	3	0	3	{"allowreopen":true,"reopenstatus":0,"description"...	2024-03-15 15:55:51	0000-00-00 00:00:00
2	Resolved	closed	1	0	2	{"allowreopen":true,"reopenstatus":0,"description"...	2024-03-15 15:55:51	0000-00-00 00:00:00
1	Open	open	3	0	1	{"description":"Open tickets."}	2024-03-15 15:55:51	0000-00-00 00:00:00

Figura 4.5: Tabela com todos os estados do *ticket*.

Foi também alterada a tabela **event** (Figura 4.6) responsável por mostrar as ações na *thread* de *tickets* presentes na interface gráfica do osTicket.

```
SELECT * FROM `ost_event` ORDER BY `ost_event`.`id` DESC
```

id	name	description
22	suspended	NULL
4	assigned	NULL
3	reopened	NULL
2	closed	NULL
1	created	NULL

Figura 4.6: Tabela com todos os eventos do *ticket*.

Como podemos ver na Figura 4.6, na interface do osTicket é agora possível observar quando um agente muda o estado do *ticket* para suspenso.

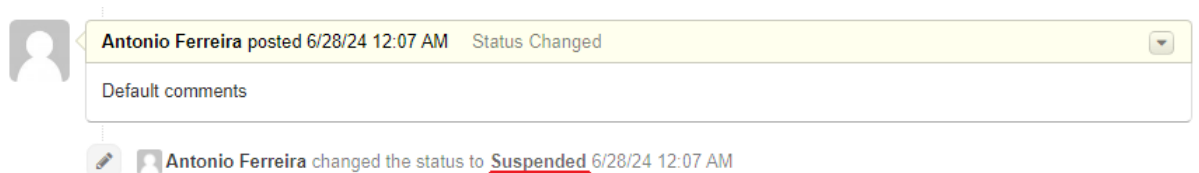


Figura 4.7: Exemplo na interface de *ticket* suspenso por agente.

4.2 Classes

Para implementar o *plugin* e todas as suas funcionalidades, foi necessário criar novas classes e utilizar classes já implementadas pelo osTicket anteriormente. Assim sendo, na Figura 4.8, pode ser observado o diagrama geral de classes do *plugin* implementado.

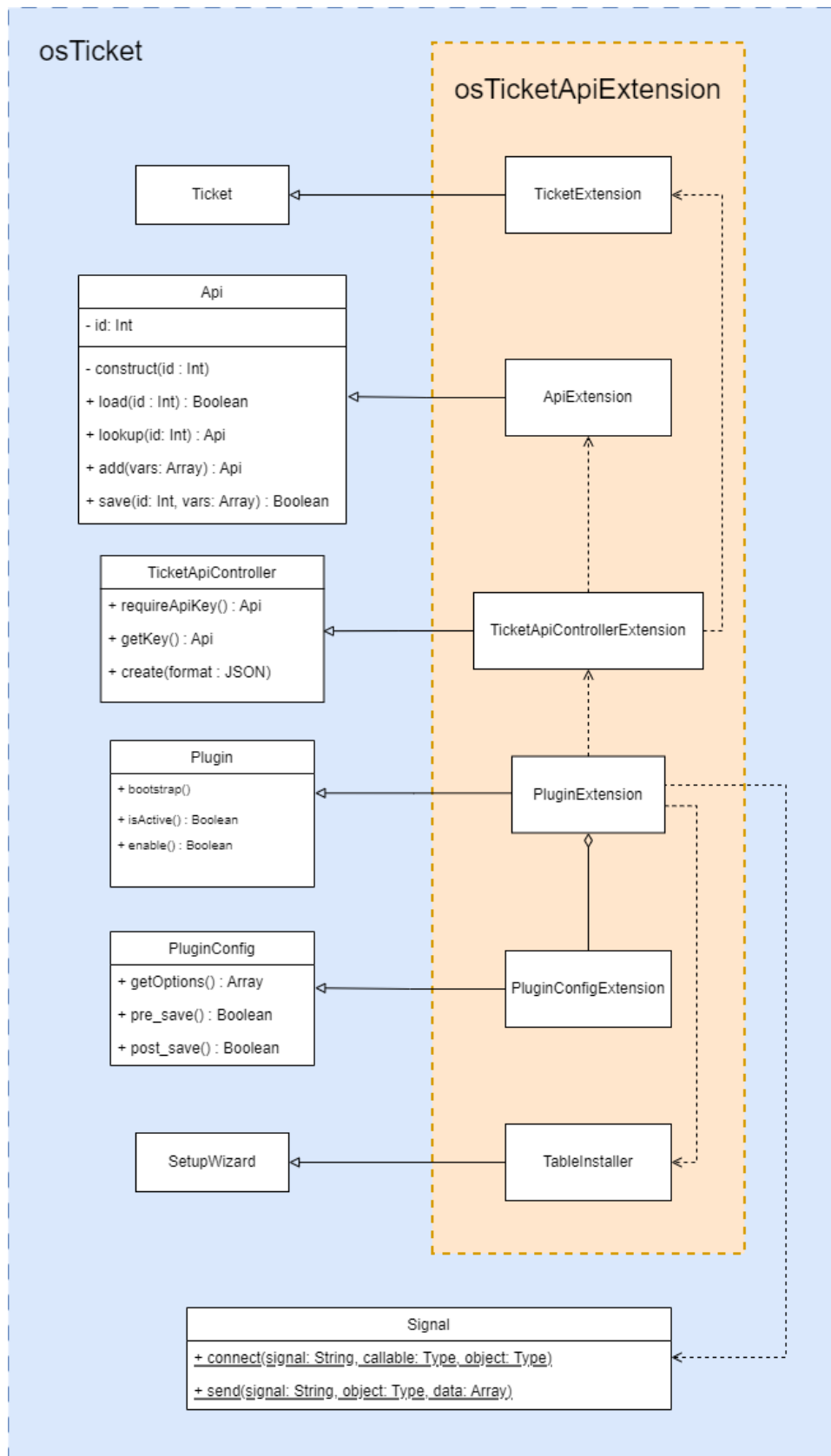


Figura 4.8: Diagrama de classes geral.

A Classe `PluginExtension`, Figura 4.9, estende da Classe `Plugin` do `osTicket`. Esta é responsável por registar os `endpoints` da API, chamar a classe que cria as tabelas extras (`table.installer.php`) e verificar o estado do *plugin* (ativo ou desativo). Caso o *plugin* seja desativado, esta classe permite também guardar as tabelas extra geradas num ficheiro *Structured Query Language* (SQL) de forma a que possam ser usadas de novo numa instalação futura do *plugin*. Isto tudo acontece através do método `bootstrap`, que é o método chamado quando o *plugin* está ativo, assim como já tinha sido referido na Secção 3.6.

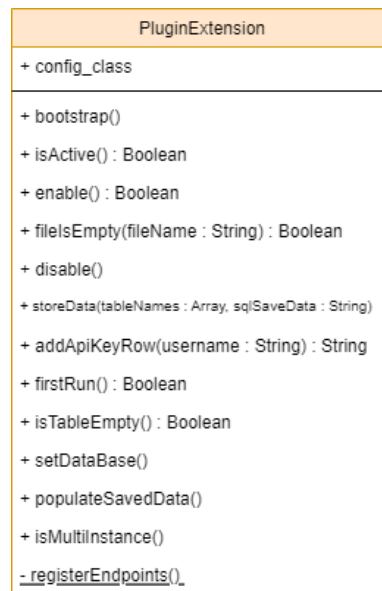


Figura 4.9: Diagrama da classe `PluginExtension`

A Classe `TableInstaller`, Figura 4.10 estende da Classe `SetupWizard`, esta classe é responsável por executar ficheiros SQL, através da função `runJob`.

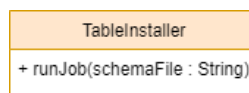


Figura 4.10: Diagrama da classe `TableInstaller`

A Classe `PluginConfigExtension`, Figura 4.11, estende da Classe `PluginConfig` do `osTicket`, e é responsável por criar o formulário com todas as configurações

do *plugin*, sendo que as configurações existentes são, uma caixa de texto onde se indica o nome do utilizador que vai obter a primeira chave da API, mais uma caixa de texto onde a chave gerada anteriormente é mostrada, e por fim, uma **checkbox** onde se indica se é pretendido guardar a informação das novas tabelas, mostradas na Secção 4.1, quando o *plugin* é desinstalado. As configurações podem ser observadas numa figura da Secção 4.4. Esta classe também é responsável por guardar os valores de configuração inseridos pelo utilizador no formulário anterior, para depois serem utilizados na classe `PluginExtension`, Figura 4.11.

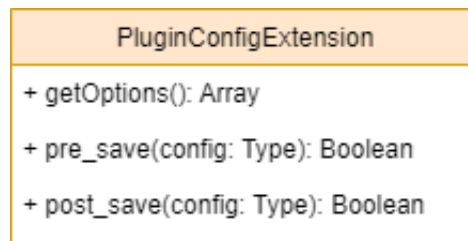
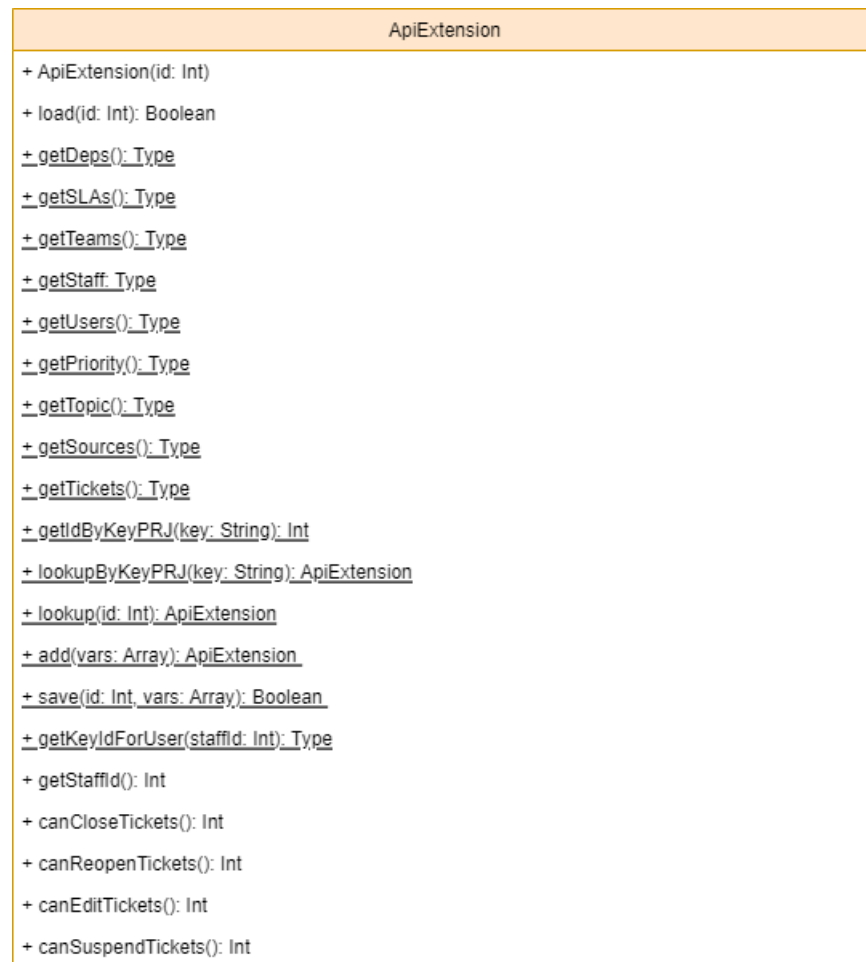


Figura 4.11: Diagrama da Classe `PluginConfigExtension`

A Classe `ApiExtension`, Figura 4.12, estende da Classe `API` do `osTicket`, e reescreve alguns dos métodos da sua classe mãe de forma a ser compatível com o novo sistema de autenticação, adiciona também verificações para adaptar a novas necessidades criadas pela nova tabela `api_key_extension`, como por exemplo verificar se o agente passado tem permissão de fechar *tickets*. Por fim é implementada a lógica dos pedidos `GET`, que consistem em executar uma *query* SQL e criar a resposta que é enviada para o utilizador que fez o pedido.

Figura 4.12: Diagrama Classe **ApiExtension**

A Classe **TicketApiControllerExtension**, Figura 4.13, estende da Classe **TicketApiController** do `osTicket` e é nesta classe que estão os métodos associados aos vários **endpoints** da API. Em todos os métodos é verificada a chave enviada no cabeçalho do pedido para garantindo que o utilizador tem as permissões necessárias para utilizar essa funcionalidade. Para além desta autenticação, nos métodos dos **endpoints** do tipo **GET**, é feita uma *query* para obter as informações da base de dados pretendidas. Nos métodos do tipo **POST**, o corpo da mensagem, independentemente do seu formato ser JSON, XML ou email, é transformado num *array*. Este *array*, consoante o de pedido feito, vai ser utilizado para realizar a operação pretendida, sendo ela fechar *ticket*, suspender *ticket*, etc. No final todos os métodos devolvem

uma resposta onde são indicados os erros encontrados, alterações feitas ou resultados obtidos. Alguns métodos da classe `TicketApiController`, que esta classe estende, foram sobrescritos (`overridden`) para serem compatíveis com a nova autenticação adotada, referida na Secção 3.3.



Figura 4.13: Diagrama da Classe `TicketApiControllerExtension`

A Classe `TicketExtension`, Figura 4.17, que estende da Classe `Ticket`,

adiciona novas funcionalidades ao objeto *Ticket*, sendo elas:

- Converter vários comentários num só, por exemplo, na edição de múltiplos campos de um *ticket* aparece apenas na interface do osTicket uma única mensagem, como é possível observar na Figura 4.14;



Figura 4.14: Exemplo de alteração simultânea de vários parâmetros de um *ticket*

- Permitir redefinir a prioridade de um *ticket*, variando entre *Low*, *Normal*, *High* e *Emergency*, possíveis de observar na Figura 4.15;

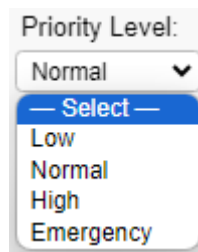


Figura 4.15: Níveis de prioridades

- Mudar o departamento associado ao *ticket*;
- Disponibilizar um mecanismo que indica a origem do *ticket* como pode ser visto na Figura 4.16. (API, telefone, *email* e outro).

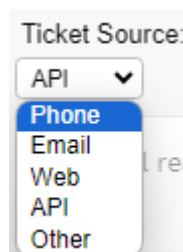


Figura 4.16: Opções de origem de *tickets*

TicketExtension
<pre> + getChanges(new: Type, old: Type): Type + editFields(field: String, newValue: Type, refer: Boolean): Boolean + addComments(comments: String, fields: Array, staffAssignee: Staff, teamAssignee: Team): Type + alerts(note: ThreadEntry) + setPriorityId(priorityId: Int): Boolean + setFormEntryValue(newValue: Int) + setSuspend(status: String, comments: String): Boolean + setDueDateToNotNull(): Boolean + <u>getSources(): Array</u> </pre>

Figura 4.17: Exemplo Classe PluginExtension

Os detalhes na íntegra de todos os métodos de cada classe podem ser verificados na respetiva documentação do trabalho, que por sua vez pode ser encontrada no Repositório online [Grupo15_online_rep, 2024].

4.3 Endpoints

Os **endpoints** são a forma da API implementada fazer uma funcionalidade diferente de acordo com o pedido do utilizador, cada funcionalidade está associada a um *Uniform Resource Locator* (URL) diferente.

Para gerar os **endpoints** através do *plugin* foi criado um novo método que é chamado no método **bootstrap**, que é o método principal do *plugin*. Desta forma sempre que o *plugin* estiver ativo os **endpoints** são criados no servidor que estiver a correr o osTicket. Na Listagem 5 pode ser observado o código do método onde se criam os novos **endpoints**, tendo em conta que só são mostrados dois **endpoints** de cada tipo de pedido como exemplo.

Listagem 5: Função registerEndpoints().

```

1 private static function registerEndpoints() {
2     $routesPOST = array(
3         array(
4             'prefix' => "open/tickets",
5             'function' => 'create'
6         ),
7         array(
8             'prefix' => "close/tickets",
9             'function' => 'close'
10        ),
11    $routesGET = array(
12        array(

```

```

13         'prefix' => "departments",
14         'function' => 'showDeps'
15     ),
16     array(
17         'prefix' => "slas",
18         'function' => 'showSLAs'
19     ),
20     foreach ($routesPOST as $route) {
21         Signal::connect('api', function ($dispatcher) use ($route) {
22             $dispatcher->append(
23                 url_post(
24                     "~/{$route['prefix']}\.(?P<format>xml|json|email)$",
25                     array(PRJ_API_DIR.'api.extension.php:TicketApiControllerExtension',
26                         $route['function'])
27                 )
28             );
29         });
30     }
31     foreach ($routesGET as $route) {
32         Signal::connect('api', function ($dispatcher) use ($route) {
33             $dispatcher->append(
34                 url_get(
35                     "~/{$route['prefix']}\.(?P<format>xml|json|email)$",
36                     array(PRJ_API_DIR.'api.extension.php:TicketApiControllerExtension',
37                         $route['function'])
38                 )
39             );
40         });
41     }

```

Como pode ser observado na Listagem 5, cada **endpoint** é guardado num array com o seu URL e o nome do método que é chamado quando é feito um pedido para o respetivo **endpoint**. Por fim, para integrar os **endpoints** no osTicket, é necessário usar o método estático da classe **Signal**, chamado **connect()**, que subscreve cada URL dos **endpoints** ao sinal **'api'**. Quando um pedido é feito para o servidor através da API, é enviado um sinal para todos os subscritores do sinal **'api'** e o método do **endpoint** correspondente ao URL utilizado para fazer o pedido é chamado. Este processo de subscrição de **endpoints** pode ser observado nas linhas 20 a 29, para métodos **POST** e das linhas 30 a 39, para métodos **GET**, da Listagem 5.

Como já tinha sido brevemente explicado na Secção 3.2, para fazer um pedido ao servidor através da API, é necessário enviar uma mensagem com um cabeçalho e corpo da mensagem se for um pedido do tipo **POST** e só o cabeçalho se for um pedido do tipo **GET**.

Em ambos os tipos de pedido, no cabeçalho é necessário enviar dois parâmetros, como pode ser observado na Figura 4.18, sendo que **JSON** pode ser alterado para **XML** ou **email** dependendo do formato do corpo. A **chave gerada** é a chave que faz a autenticação do utilizador e tem as permissões

para fazer o pedido à API.

```
{  
    "Accept": "application/json",  
    "X-API-KEY": "chave gerada"  
}
```

Figura 4.18: Cabeçalho do pedido com formato JSON

O corpo da mensagem, só é utilizado nos pedidos do tipo `POST` e varia de acordo com o `endpoint` utilizado, na Figura 4.19 pode ser observado um exemplo de um pedido ao `endpoint` para fechar *tickets*, em formato JSON.

```
{  
    "ticketNumber": "636681",  
    "comments": "Default comments"  
}
```

Figura 4.19: Cabeçalho do pedido para fechar *tickets* com formato JSON

Outro exemplo, é o `endpoint` para pedir uma nova chave ao servidor que difere em relação aos outros, pois para utilizar este `endpoint`, em vez de autenticação ser feita através de uma chave da API, é utilizado o nome e *password* do utilizador que o está a chamar, e é também verificado se o utilizador é um administrador, pois só administradores podem gerar novas chaves, como é feito no osTicket. Na Figura 4.20 pode ser observado um exemplo de como estruturar o corpo deste tipo de pedido, o cabeçalho é igual a todos os outros `endpoints`.

```
{
  "admin": "pmss",
  "adminPassword": "123123",
  "staff": "pmss",
  "isActive": "1",
  "canCreateTickets": "1",
  "canCloseTickets": "1",
  "canReopenTickets": "1",
  "canEditTickets": "1",
  "canSuspendTickets": "1",
  "notes": "notas do admin"
}
```

Figura 4.20: Cabeçalho do pedido para gerar uma nova chave com formato JSON

Todos os pedidos têm este tipo de estrutura e exemplos específicos de como chamar cada **endpoint** e o que faz cada campo no corpo das mensagens, pode ser observado em [Grupo15, 2024].

4.4 Detalhes do *plugin*

Como o *plugin* foi desenvolvido com o objetivo específico de estender a API já existente, existem algumas diferenças na forma como o *plugin* é instalado em relação ao que foi referido na Secção 3.7. Isto porque existem configurações que têm de ser feitas pelo utilizador antes de acabar a sua instalação, e novas tabelas e valores têm de ser adicionadas ou removidas da base de dados quando o *plugin* é instalado ou desinstalado.

A instalação do *plugin* é inicialmente igual à instalação de um *plugin* genérico mostrado na Secção 3.7, até à parte em que a instância do *plugin* é ativada, antes de se carregar no botão **Add Instance**, é necessário seleccionar a *tab* **Config**, como pode ser observado na Figura 4.21.

The screenshot shows the 'Add New Instance' form for the 'OSTicket API Extension (Disabled)' plugin. The 'Config' tab is selected, and the 'Status' dropdown is open, showing 'Enabled' and 'Disabled' options. Red arrows point to the 'Config' tab and the 'Enabled' option. The form includes fields for 'Name' (Exemplo), 'Status' (Disabled), and 'Internal Notes'. The 'Add Instance' button is highlighted in orange.

Plugins > OSTicket API Extension (Disabled) — Add New Instance

Instance Config ← Importante antes de carregar em "Add Instance" fazer a "config"

Instance Name and Status

Name: Exemplo 1º Nome de Exemplo

Status: Disabled

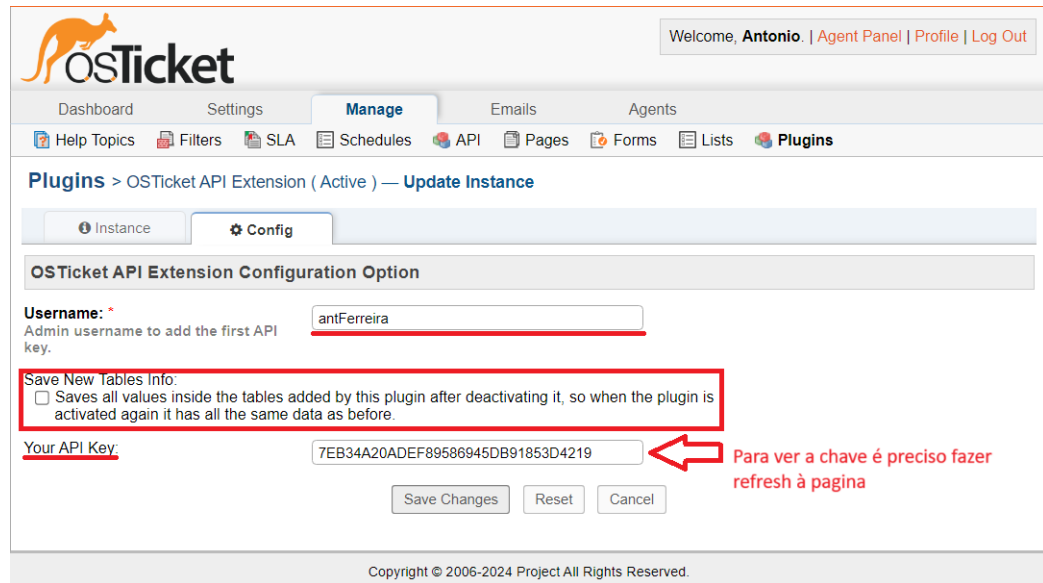
Internal Notes: Instance des notes

Add Instance Reset Cancel

Copyright © 2006-2024 Antonio usFerreira All Rights Reserved

Figura 4.21: Selecionar Config

Na *tab* Config são mostrada todas as opções de configuração do *plugin*. Aqui é necessário preencher o **username** do agente à qual a primeira chave do *plugin* vai ser atribuída, sendo este normalmente o **username** do administrador que estiver a fazer a instalação. Para verificar a chave gerada pelo utilizador é necessário fazer *refresh* à página e a chave irá aparecer na caixa de texto no final das configurações. Também é possível selecionar a **checkbox** **Save New Tables Info**, que ativa a opção de guardar todas as tabelas adicionadas pelo *plugin* se este for desinstalado. Por fim basta premir o botão **Add Instance**. Este processo está todo ilustrado na Figura 4.22.



OSTicket

Welcome, Antonio | Agent Panel | Profile | Log Out

Dashboard Settings **Manage** Emails Agents

Help Topics Filters SLA Schedules API Pages Forms Lists Plugins

Plugins > OSTicket API Extension (Active) — Update Instance

Instance Config

OSTicket API Extension Configuration Option

Username: * antFerreira
Admin username to add the first API key.

Save New Tables Info:
☐ Saves all values inside the tables added by this plugin after deactivating it, so when the plugin is activated again it has all the same data as before.

Your API Key: 7EB34A20ADEF89586945DB91853D4219

Save Changes Reset Cancel

Para ver a chave é preciso fazer refresh à pagina

Copyright © 2006-2024 Project All Rights Reserved.

Figura 4.22: Selecionar Opções de configuração do *plugin*

Se o *plugin* tiver sido instalado corretamente, irá ser recebida uma notificação de sucesso na interface do osTicket, como mostrado no fim da Secção 3.7. As novas tabelas e valores referidos na Secção 4.1 são inseridos na base de dados, os *scripts* de instalação destas tabelas e valores encontram-se no ficheiro `sql/scripts.sql`.

Depois da instalação do *plugin* estar concluída é necessário garantir que os `ids` das linhas da tabela `ticket_status` e os seus respetivos estados, correspondem aos valores dos estados dos *tickets* encontrados no ficheiro `plugin.config.php`, apresentado na Listagem 6.

Listagem 6: Ficheiro `plugin.config.php`

```

1
2 // Define states constants.
3 define('STATE_OPEN', 1);
4 define('STATE_RESOLVE', 2);
5 define('STATE_CLOSE', 3);
6 define('STATE_ARCHIVED', 4);
7 define('STATE_RESOLVED', 5);
8 define('STATE_SUSPENDED', 6);
9
10 // Define names of database tables.
11 define('API_NEW_TABLE', TABLE_PREFIX . 'api_key_extension'); // Table for
    new API keys.
12 define('SUSPEND_NEW_TABLE', TABLE_PREFIX . 'suspended_ticket'); // Table
    for suspended tickets.
13
14 // Define directories to SQL files.

```

```

15 define('PRJ_PLUGIN_DIR', INCLUDE_DIR . 'plugins/OSTicket-API-Extension/');
    // Plugin directory.
16 define('SQL_SCRIPTS_DIR', PRJ_PLUGIN_DIR . 'sql/'); // Directory for SQL
    scripts.
17 define('PRJ_API_DIR', PRJ_PLUGIN_DIR . 'api/'); // Directory for API files.
18
19 // Define names of SQL files.
20 define('SAVED_DATA_SQL', SQL_SCRIPTS_DIR . 'savedData.sql'); // SQL file
    for saved data.
21 define('UNINSTALL_SCRIPT', SQL_SCRIPTS_DIR . 'uninstallScript.sql'); // SQL
    file for uninstallation.
22 define('INSTALL_SCRIPT', SQL_SCRIPTS_DIR . 'scripts.sql'); // SQL file for
    installation.

```

A desinstalação do *plugin* desenvolvido também tem um passo a mais que a desinstalação de um *plugin* genérico mostrada na Secção 3.7. No caso específico do *plugin* desenvolvido, antes de se seleccionar a opção **delete**, é necessário desativar primeiro o *plugin*, como mostra a Figura 4.23, verificar se foi desativado como podemos ver na Figura 4.24, e só depois seleccionar a opção **delete** como podemos observar na Figura 4.25.

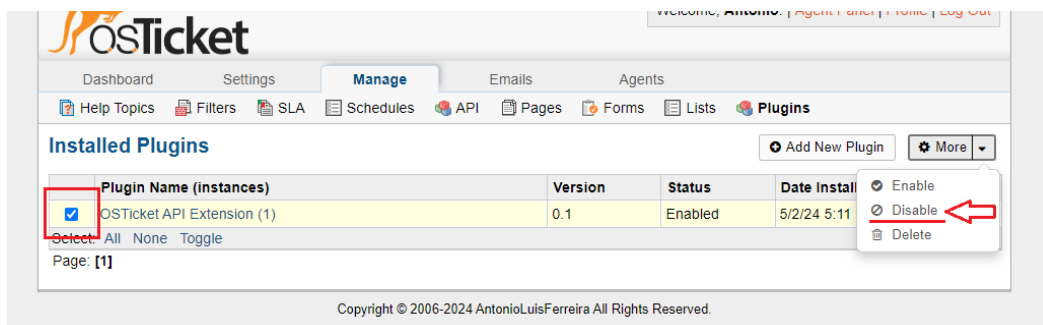


Figura 4.23: Desativar o *plugin*

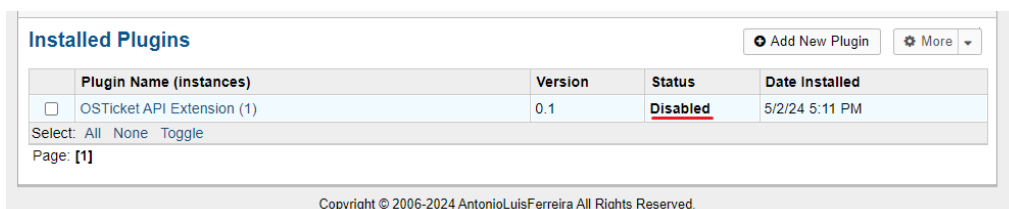
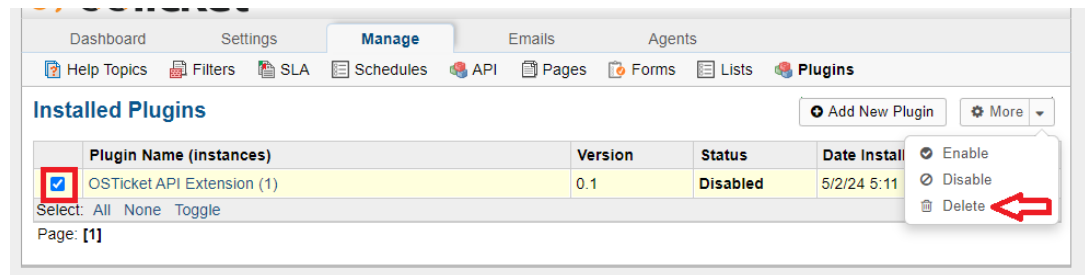


Figura 4.24: Verificar que o *plugin* está desativado

Figura 4.25: Apagar o *plugin*

Este passo extra em relação à desinstalação de um *plugin* genérico é necessário, pois a remoção das novas tabelas e valores, mostrados na Secção 4.1, só é possível de ser feita quando um *plugin* é desativado, devido à forma como os *plugins* estão estruturados no osTicket. Os *scripts* de desinstalação das novas tabelas e valores encontram-se no ficheiro `sql/uninstallScript.sql`.

Se a opção **Save New Tables Info** das configurações do *plugin* desenvolvido estiver seleccionada, quando o *plugin* é desativado, antes de as novas tabelas serem removidas da base de dados, toda a sua informação é guardada no ficheiro `sql/savedData.sql`. Quando o *plugin* volta a ser instalado, depois de as tabelas terem sido outra vez criadas, os *scripts* dentro deste ficheiro são executados e preenchem as tabelas com os valores que tinham durante a ultima instalação.

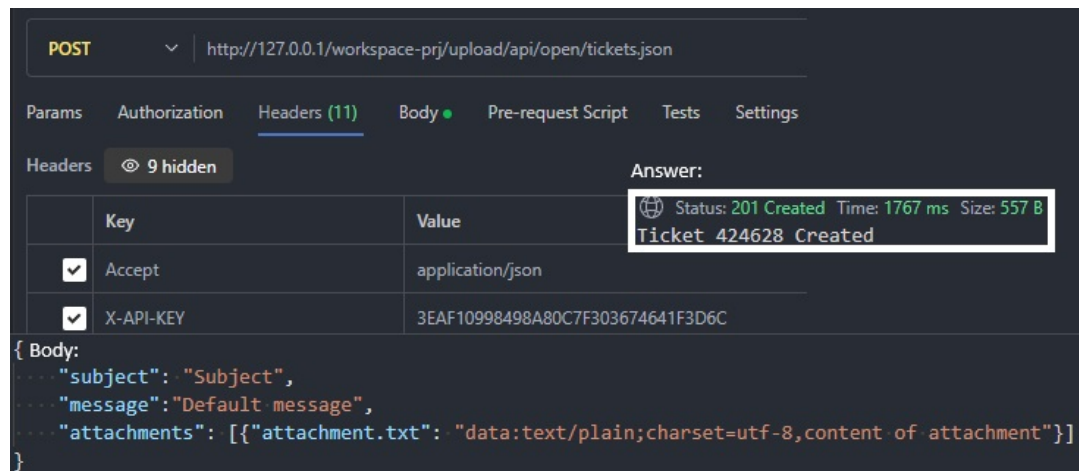
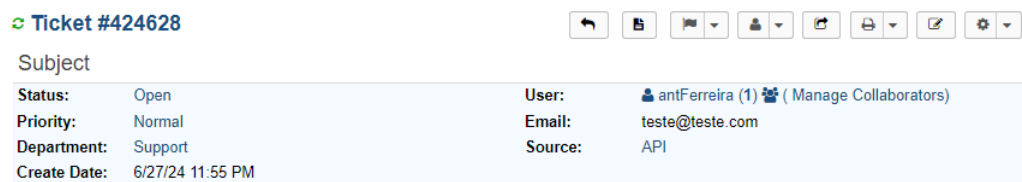
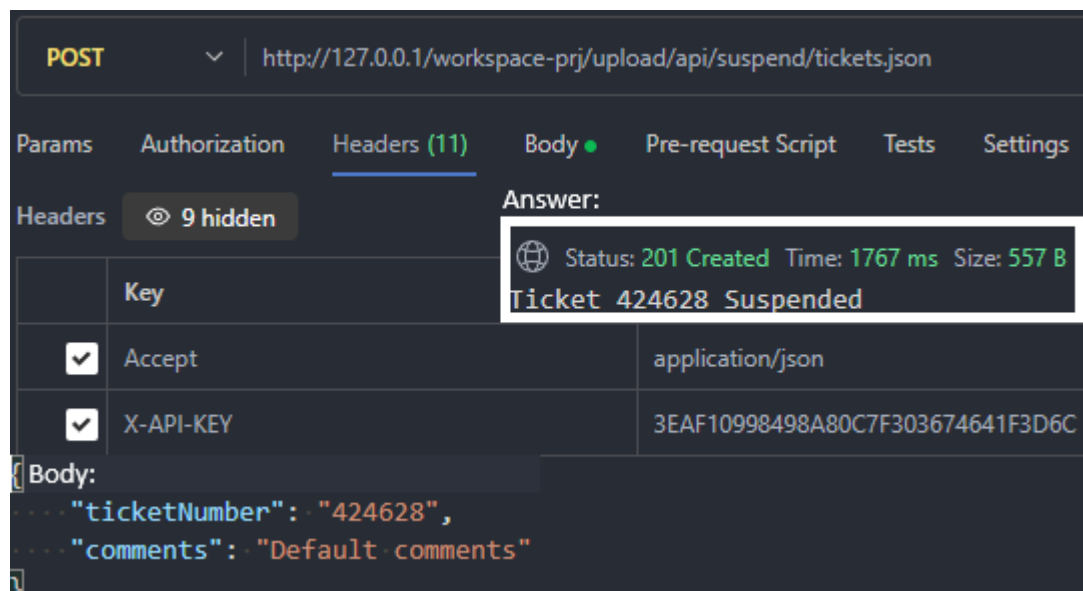
Capítulo 5

Validação e Testes

Para garantir que a API desenvolvida para o osTicket tenha o comportamento esperado, foram realizados testes de validação usando o *software* Postman, uma ferramenta de testes para a API.

O ambiente de desenvolvimento foi configurado num **container Docker** que usa como sistema operativo **Linux**. Dentro deste ambiente, utilizou-se a versão 8.1.2 do **XAMPP**, que integra a versão 10.4.22 do **MySQL** e 8.0 de **PHP**, fornecendo uma plataforma estável e consistente para o desenvolvimento e testes.

Todos os **endpoints** da API foram testados. É possível observar nas Figuras 5.1, 5.3 e 5.5, a boa utilização de três **endpoints** dos **endpoints** enunciados na Listagem 5 no Secção 4.3, com as vistas da interface disponibilizada pelo osTicket nas Figuras 5.2, 5.4 e 5.6 respetivamente.

Figura 5.1: Pedido de abrir *ticket* no *software* PostmanFigura 5.2: Interface do osTicket com o *ticket* no estado aberto.Figura 5.3: Pedido de suspender *ticket* no *software* Postman

Ticket #424628

Subject

Status:	Suspended	User:	antFerreira (1) (Manage Collaborators)
Priority:	Normal	Email:	teste@teste.com
Department:	Support	Source:	API
Create Date:	6/27/24 11:55 PM		

Assigned To:	Antonio Ferreira	Help Topic:	None
SLA Plan:	Default SLA	Last Message:	6/27/24 11:55 PM
Due Date:	7/2/24 8:00 AM	Last Response:	

Figura 5.4: Interface do osTicket com o *ticket* no estado suspenso.

POST `http://127.0.0.1/workspace-prj/upload/api/close/tickets.json`

Params Authorization Headers (11) Body ● Pre-request Script Tests Settings

Headers 9 hidden

Answer:

Status: 201 Created Time: 1767 ms Size: 557 B
Ticket 424628 Closed

Body:

```
{
  "ticketNumber": "424628",
  "comments": "Default comments"
}
```

Figura 5.5: Pedido de fechar *ticket* no *software* Postman

Ticket #424628

Subject

Status:	Closed	User:	antFerreira (1) (Manage Collaborators)
Priority:	Normal	Email:	teste@teste.com
Department:	Support	Source:	API
Create Date:	6/27/24 11:55 PM		

Closed By:	Antonio Ferreira	Help Topic:	None
SLA Plan:	Default SLA	Last Message:	6/27/24 11:55 PM
Close Date:	6/28/24 2:01 PM	Last Response:	

Figura 5.6: Interface do osTicket com o *ticket* no estado fechado.

O resultado obtido foi positivo em todos os testes realizados, no Apêndice

A está o ficheiro JSON que após ser importado para o *software* **Postman** testa todas as funcionalidades da API.

Capítulo 6

Conclusões e Trabalho Futuro

A realização deste Projeto de desenvolvimento de uma API que permite, por via programática, realizar ações disponibilizadas pela interface do osTicket (abrir *tickets*, fechar *tickets*, etc) demonstrou ser uma iniciativa de grande relevância e impacto visto que, até hoje, não existe disponível *online* qualquer API com funcionalidades semelhantes. Foi também desenhado e implementado um novo mecanismo de suspender *tickets* que até esta versão do osTicket, não foi disponibilizado.

Ao tornar o método de desenvolvimento de uma API e do respetivo *plugin* que a instala disponíveis para a comunidade, é aberto um caminho para que outras pessoas e organizações possam desenvolver as suas próprias API e respetivos *plugins* personalizados de acordo com as suas necessidades específicas.

É também importante referir que, o *plugin* desenvolvido, visto que não altera código original do *software* osTicket, é compatível com qualquer outro *plugin* que siga também uma estrutura modular que não altere código original.

Como trabalho futuro, existem algumas vertentes que podem ser exploradas, como por exemplo a criação de Departamentos, Agentes, campos específicos de um *ticket* entre outros, via API.

Por fim, todas as classes que formam o *plugin* que instala a API, a sua documentação e o seu manual de instalação e utilização estão disponíveis online ([Grupo15, 2024]) tal como o manual de criação de um *plugin* genérico no Apêndice B.

Apêndice A

Criação de um *Plugin* na Íntegra

28/06/24, 13:19

criar_plugin_osticket

Desenvolvimento de um Plugin para o osTicket 1.18

1.

Todos os plugins do osTicket devem estar na diretoria <INSTALL_ROOT>/include/plugins.

Cada plugin deve estar em pastas separadas, por exemplo [INSTALL_ROOT]/include/plugins/myplugin

Tem de haver dois ficheiros obrigatoriamente necessários para que o OSTicket reconheça o plugin:

- plugin.php
- config.php

plugin.php:

Este ficheiro contém uma descrição geral do plugin, segue então um exemplo:

```
<?php
return array(
    set_include_path(get_include_path() . PATH_SEPARATOR . dirname(__file__) . '/include'),
    'id' => 'proposal-15:extension',
    'version' => '0.1',
    'name' => 'OSTicket API Extension',
    'author' => 'Grupo Proposta 15',
    'description' => 'Adds extra functionality to OSTicket API.',
    'plugin' => 'extension.php:PluginExtension'
);
?>
```

A sintax é 'file name:class name', sendo a parte mais importante deste ficheiro a linha:

```
'plugin' => 'extension.php:PluginExtension'
```

visto que é a classe que adiciona as funcionalidades que não existem, no nosso caso, instala tabelas e cria novos endpoints.

28/06/24, 13:19

criar_plugin_osticket

2.

O backend do admin consegue mostrar as opções de configuração de instâncias do plugin. Estas estão definidas no ficheiro config.php

É aconselhado a criação de um ficheiro à parte para a definição de constantes que possam ser utilizadas no futuro, como por exemplo:

```
// Define names of database tables.
define('API_NEW_TABLE', TABLE_PREFIX . 'api_key_extension'); // Table for new API keys.
define('SUSPEND_NEW_TABLE', TABLE_PREFIX . 'suspended_ticket'); // Table for suspended tickets.

// Define directories to SQL files.
define('PRJ_PLUGIN_DIR', INCLUDE_DIR . 'plugins/OSTicket-API-Extension/'); // Plugin directory.
define('SQL_SCRIPTS_DIR', PRJ_PLUGIN_DIR . 'sql/'); // Directory for SQL scripts.
define('PRJ_API_DIR', PRJ_PLUGIN_DIR . 'api/'); // Directory for API files.
define('PRJ_UTIL_DIR', PRJ_PLUGIN_DIR . 'util/'); // Directory for Util files.

// Define names of SQL files.
define('SAVED_DATA_SQL', SQL_SCRIPTS_DIR . 'savedData.sql'); // SQL file for saved data.
define('UNINSTALL_SCRIPT', SQL_SCRIPTS_DIR . 'uninstallScript.sql'); // SQL file for uninstalatio
define('INSTALL_SCRIPT', SQL_SCRIPTS_DIR . 'scripts.sql'); // SQL file for installation.
```

Este ficheiro auxiliar pode então ser importado para o ficheiro config.php

Segue então um exemplo do ficheiro config.php:

28/06/24, 13:19

criar_plugin_osticket

```

<?php

require_once(INCLUDE_DIR.'/class.forms.php');

class PluginConfigExtension extends PluginConfig {

    function getOptions() {
        list($__, $_N) = self::translate();
        return array(
            'title' => new SectionBreakField(array(
                'label' => $__('OSTicket API Extension Configuration Option'),
            )),
            'username' => new TextboxField(array(
                'label' => __('Username'),
                'required' => true,
                'configuration' => array('size'=>40),
                'hint' => __('Admin username to add the first API key.'),
            )),
            'save_info' => new BooleanField(array(
                'id' => 'save_info',
                'label' => __('Save New Tables Info'),
                'default' => false,
                'configuration' => array(
                    'desc' => __('Saves all values inside the tables added by this plugin after de
                    so when the plugin is activated again it has all the same data as before.')
                )
            )),
            'apikey' => new TextboxField(array(
                'label' => $__('Your API Key'),
                'configuration' => array('size'=>40),
            )),
        );
    }

    function pre_save(&$config, &$errors) {
        global $msg;

        if (!$errors)
            $msg = __('Configuration updated successfully');
        return true;
    }

    public function post_save(&$config, &$errors) {

```

28/06/24, 13:19

criar_plugin_osticket

```

global $msg;

// Verifica se a opção 'save_on_deactivate' foi selecionada
if (isset($config['save_info'])) {
    $this->set('save_info', $config['save_info']);
    $msg = 'Save on deactivate setting updated.';
}
return true;
}
}

```

Esta classe estende 'PluginConfig' localizada em [INSTALL_ROOT]/include/class.plugin.php (IMPORTANTE)

A função getOptions() retorna um array com as opções de configuração do plugin.

```

'title' => new SectionBreakField(array(
    'label' => __('OSTicket API Extension Configuration Option'),
)),
'username' => new TextboxField(array(
    'label' => __('Username'),
    'required' => true,
    'configuration' => array('size'=>40),
    'hint' => __('Admin username to add the first API key.'),
)) ...

```

Este exemplo define a configuração das variáveis 'title' e 'username'.

Neste caso, a variável 'title' vai ser representada por um título na página com o texto 'OSTicket API Extension Configuration Option' e a variável 'username' representa uma caixa de texto que receberá o nome de utilizador do administrador.

A seguinte lista mostra todos os campos disponíveis, definidos em [INSTALL_ROOT]/include/class.forms.php:

28/06/24, 13:19

criar_plugin_osticket

TextboxField - text box
TextareaField - text area
ThreadEntryField - rich text area, used for discussion threads
DatetimeField - JQuery datepicker
PhoneField - text box optimized for phone numbers
BooleanField - checkbox
ChoiceField - drop-down select field
SectionBreakField - horizontal section break

```
function pre_save(&$config, &$errors)
```

Esta função vai ser chamada quando a configuração for guardada pelo administrador.

3.

Ficheiro 'extension.php'

(referido anteriormente em: 'plugin' => 'extension.php:PluginExtension')

Dentro deste ficheiro estará a lógica toda do plugin, desde a instalação das tabelas até à criação de funções que são chamadas nos endpoints

Esta nova classe criada terá obrigatoriamente de estender a classe Plugin definida no ficheiro

[INSTALL_ROOT]/include/class.plugin.php

28/06/24, 13:19

criar_plugin_osticket

```
<?php

include_once 'plugin.config.php';
require_once 'class.staff.php';
require_once 'class.plugin.php';
require_once 'api/class.api.extension.php';
require_once 'util/table.installer.php';
require_once 'config.php';

include INCLUDE_DIR . 'class.dispatcher.php';

class PluginExtension extends Plugin
{

    //referencia para a classe das configuracoes do plugin
    var $config_class = 'PluginConfigExtension';

    function bootstrap()
    {
        $config = $this->getConfig();
        $username = $config->get('username');

        self::registerEndpoints();

        if($this->isTableEmpty(API_NEW_TABLE)){
            $key = $this->addApiKeyRow($username);
            $config->set('apikey', $key);
        }
    }

    function isActive()
    {
        if (!parent::isActive()) {
            $this->disable();
        } else {
            $this->enable();
        }
        return parent::isActive();
    }

    function enable()
    {
        $saveInfo = false;
```

28/06/24, 13:19

criar_plugin_osticket

```

$instances = $this->getActiveInstances();
foreach ($instances as $instance) {
    $saveInfo = $instance->getConfig()->get('save_info');
}

if($this->firstRun()){
    $this->setDataBase();
    if($saveInfo){
        $this->populateSavedData();
    }
}

return parent::enable();
}

//verifica se existe um ficheir com os valores da tabvela guardados
function fileIsEmpty($filename){

    if (file_exists($filename)) {

        $handle = fopen($filename, "r");
        $filesize = filesize($filename);
        fclose($handle);

        //verifica se o ficheiro tem alguma informacao la dentro baseado no seu t:
        if ($filesize > 0) {
            return false; //nao esta vazio, logo vai correr o que esta la dent
        } else {
            return true; //esta vazio, o plugin corre por defualt com uma api
        }
    } else {
        return true; //o ficheiro nao existe
    }
}

function disable()
{
    if($this->firstRun()){
        return;
    }

    //ve se o utilizador quer guardar a informacao das tabelas ou nao, true por defau:
    $saveInfo = false;

```

28/06/24, 13:19

criar_plugin_osticket

```

$instances = $this->getActiveInstances();
foreach ($instances as $instance) {
    $saveInfo = $instance->getConfig()->get('save_info');
}

if($saveInfo){
    //guarda a informacao das novas tabelas num ficheiro sql
    //suporta varias tabelas, se criarmos novas é so adicionar o nome a array
    $tableNames = array(
        API_NEW_TABLE,
        SUSPEND_NEW_TABLE
        //ADICIONAR NOVAS TABELAS AQUI
    );
    $this->storeData($tableNames, SAVED_DATA_SQL);
}
//desinstala as tabelas e linhas novas da base de dados
$installer = new TableInstaller();
$installer->runJob(UNINSTALL_SCRIPT);
}

function storeData($tableNames, $sqlSaveData){
    //guarda os resultados de cada tabela numa array
    $tablesArray = array();

    foreach($tableNames as $tableName){
        //exporta a informação para um ficheiro
        $sql_query = 'SELECT * FROM `'. $tableName . '`';

        $res = db_query($sql_query);
        /* db_query($sql_query); */

        //vai buscar as linhas todas -> informação
        $array = db_assoc_array($res);

        //guarda o resultado desta tabela na array de tabelas
        $tablesArray[$tableName] = $array;
    }

    //guarda os inserts todos no ficheiro sql
    $file = fopen($sqlSaveData, 'w');

    //warning no sql para nao ser alterado
    $warning = "-- DON'T CHANGE THIS FILE UNLESS YOU KNOW WHAT YOU'RE DOING!!!\n\n";

```

28/06/24, 13:19

criar_plugin_osticket

```

fwrite($file, $warning);

foreach($tablesArray as $tableName => $array){

    //se a array esta vazia é porque essa tabela esta vazia entao da skip
    if (!empty($array)) {
        $columns = array_keys($array[0]);
    } else {
        continue;
    }

    $columns = array_keys($array[0]);
    $valuesArrays = array_fill(0, count($columns), array());

    foreach ($array as $arr) {
        foreach ($columns as $index => $column) {
            $valuesArrays[$index][] = "" . addslashes($arr[$column])
        }
    }

    $insertSQL = "INSERT INTO `{$tableName}` (" . implode(", ", array_map(function($column) {
        return "`{$column}`";
    }, $columns)) . ")\n VALUES ";

    foreach ($valuesArrays[0] as $rowIndex => $value) {
        $insertSQL .= "(" . implode(", ", array_column($valuesArrays, $rowIndex)) . "\n";
    }

    $insertSQL = rtrim($insertSQL, ",\n") . ";\n\n";

    fwrite($file, $insertSQL);
}

fclose($file);
}

function addApiKeyRow($username)
{
    $staff = Staff::lookup($username);
    // $staff = StaffAuthenticationBackend::getUser(); tentativa de meter o staff com login c

    $data = array(

```


28/06/24, 13:19

criar_plugin_osticket

```

        'idStaff' => "{$staff->getId()}",
        'isActive' => "1",
        'canCreateTickets' => "1",
        'canCloseTickets' => "1",
        'canReopenTickets' => "1",
        'canEditTickets' => "1",
        'canSuspendTickets' => "1",
        'notes' => "An API key automatically generated upon the plugin first run."
    );

    ApiExtension::add($data, $erros);
    $apikey = ApiExtension::lookup(1);
    return $apikey->getKey();
}

function firstRun()
{
    $sql = 'SHOW TABLES LIKE \'\' . API_NEW_TABLE . \'\'';
    $res = db_query($sql);
    return (db_num_rows($res) == 0);
}

function isTableEmpty($tableName)
{
    //limit 1 para ir buscar so a primeira linha em vez todas, porque podem existir m:
    $sql = 'SELECT * FROM \'\' . $tableName . \'\' LIMIT 1';
    $res = db_query($sql);
    return (db_num_rows($res) == 0);
}

function setDataBase()
{
    $installer = new TableInstaller();
    $installer->runJob(INSTALL_SCRIPT);
}

function populateSavedData(){
    $installer = new TableInstaller();
    $installer->runJob(SAVED_DATA_SQL);
}

function isMultiInstance(){
    return false;
}

```

28/06/24, 13:19

criar_plugin_osticket

```
}

private static function registerEndpoints()
{

    $routesPOST = array(
        array(
            'prefix' => "open/tickets",
            'function' => 'create'
        ),
        array(
            'prefix' => "close/tickets",
            'function' => 'close'
        ),
        array(
            'prefix' => "suspend/tickets",
            'function' => 'suspend'
        ),
        array(
            'prefix' => "requestApiKey/tickets",
            'function' => 'requestApiKey'
        ),
        array(
            'prefix' => "reopen/tickets",
            'function' => 'reopen'
        ),
        array(
            'prefix' => "edit/tickets",
            'function' => 'edit'
        ),
        array(
            'prefix' => "getApiKey/tickets",
            'function' => 'getUserApiKey'
        )
    );

    $routesGET = array(
        array(
            'prefix' => "departments",
            'function' => 'showDeps'
        ),
        array(
            'prefix' => "slas",
```

28/06/24, 13:19

criar_plugin_osticket

```

        'function' => 'showSLAs'
    ),
    array(
        'prefix' => "teams",
        'function' => 'showTeams'
    ),
    array(
        'prefix' => "staff",
        'function' => 'showStaff'
    ),
    array(
        'prefix' => "users",
        'function' => 'showUsers'
    ),
    array(
        'prefix' => "priorities",
        'function' => 'showPriority'
    ),
    array(
        'prefix' => "topics",
        'function' => 'showTopic'
    ),
    array(
        'prefix' => "sources",
        'function' => 'showSources'
    )
);

foreach ($routesPOST as $route) {
    Signal::connect('api', function ($dispatcher) use ($route) {
        $dispatcher->append(
            url_post(
                "^/{$route['prefix']}\.(?P<format>xml|json|email);
                array(PRJ_API_DIR, 'api.extension.php:TicketApiCon
            )
        );
    });
}

foreach ($routesGET as $route) {
    Signal::connect('api', function ($dispatcher) use ($route) {
        $dispatcher->append(

```

28/06/24, 13:19

```

                                criar_plugin_osticket
url_get(
    "^/{$route['prefix']}\.(?P<format>xml|json|email);
    array(PRJ_API_DIR. 'api.extension.php:TicketApiCont
    )
);
});
}
}
}
}

```

A função `bootstrap()` é chamada sempre que o plugin é inicializado.

O plugin precisa de estar instalado e ativo e a sua instância também para esta função funcionar.

Obtem todas as configurações não guardadas na função `init()`

Regista todos os endpoints da API.

Caso seja a primeira instalação, cria as tabelas e linhas necessárias.

```

function bootstrap()
{
    $config = $this->getConfig();
    $username = $config->get('username');

    self::registerEndpoints();

    if($this->isTableEmpty(API_NEW_TABLE)){
        $key = $this->addApiKeyRow($username);
        $config->set('apikey', $key);
    }
}

```

A função `disable()` é chamada quando existe uma verificação do `osTicket` para ver se o plugin está ativo e ele não está.

Esta é a forma de chamar código do plugin quando é desativado. Mas é preciso ter cuidado pois o `disable` é chamado várias vezes enquanto o plugin está desativado.

A função `enable()` é igual há função `disable` mas quando o plugin está ativado. Dá override à função já existente da classe `plugin`.

A função `isActive` é a função que é chamada pelo `osTicket` para verificar se o plugin está ativo ou não.

É dentro desta função que as funções `disable()` e `enable()` são chamadas. Para as duas funções

28/06/24, 13:19

criar_plugin_osticket

poderem ser chamadas esta função também é overridden da original.

Assim é possível criar código que só é corrido quando o plugin está ativado e vice-versa.

```
function bootstrap()  
    function isActive()  
    {  
        if (!parent::isActive()) {  
            $this->disable();  
        } else {  
            $this->enable();  
        }  
        return parent::isActive();  
    }  
}
```

Para criar endpoints para serem utilizados na API foi criada a função `registerEndpoints()`.

Esta função é chamada no bootstrap, sempre que o plugin está ativado.

Para criar endpoints basta adicionar novos endpoints a cada array.

Cada novo pedido tem um prefixo e uma função, sendo o prefixo o nome do pedido no url sendo o url `{DOMÍNIO}/api/(NomeDoPedido)` e a função o nome da função que o pedido irá chamar.

Para dizer onde é que a função se encontra é necessário indicar a classe, ficheiro php e diretoria onde a função se encontra, como pode ser verificado no exemplo.

No nosso exemplo existe uma array para pedidos do tipo Post e uma para Get.

Para se ter outro tipo de pedidos tem de se criar um novo `foreach()` que utilize o tipo de url que se queira.

28/06/24, 13:19

criar_plugin_osticket

```
private static function registerEndpoints()
{
    $routesPOST = array(
        array(
            'prefix' => "open/tickets",
            'function' => 'create'
        ),
        array(
            'prefix' => "close/tickets",
            'function' => 'close'
        )
    );

    foreach ($routesPOST as $route) {
        Signal::connect('api', function ($dispatcher) use ($route) {
            $dispatcher->append(
                url_post( //ALTERAR AQUI TIPO DE URL SE FOR NECESSÁRIO
                    "^/{$route['prefix']}\.(?P<format>xml|json|email);$"
                    array(PRJ_API_DIR.'api.extension.php:TicketApiCont
                )
            );
        });
    }
}
```

Referências

Caso sejam encontradas algumas dificuldades na criação do plugin é aconselhado visitar o guia pelo nos baseamos:

<https://github.com/poctob/OSTEquipmentPlugin/wiki/Plugin-Development-Introduction>

Apêndice B

Testes Postman

```
1 {
2   "info": {
3     "_postman_id":
4       "b2e68204-e37f-428c-8936-68cdb35c8214",
5     "name": "API-tests",
6     "schema":
7       "https://schema.getpostman.com/json/collection/v2.1.0/collection.json",
8     "_exporter_id": "34224011"
9   },
10  "item": [
11    {
12      "name": "Gets",
13      "item": [
14        {
15          "name": "Departments",
16          "protocolProfileBehavior": {
17            "disableBodyPruning": true
18          },
19          "request": {
20            "method": "GET",
21            "header": [
22              {
23                "key": "Cache-Control",
24                "value": "no-cache",
25                "name": "cache-control",
26                "type": "text"
27              },
28              {
29                "key": "Postman-Token",
```

```
28         "value": "<calculated when request is
29             sent>",
30         "name": "postman-token",
31         "type": "text"
32     },
33     {
34         "key": "Content-Type",
35         "value": "application/json",
36         "name": "content-type",
37         "type": "text"
38     },
39     {
40         "key": "Content-Length",
41         "value": "<calculated when request is
42             sent>",
43         "name": "content-length",
44         "type": "text"
45     },
46     {
47         "key": "Host",
48         "value": "<calculated when request is
49             sent>",
50         "name": "host",
51         "type": "text"
52     },
53     {
54         "key": "User-Agent",
55         "value": "PostmanRuntime/7.39.1",
56         "name": "user-agent",
57         "type": "text"
58     },
59     {
60         "key": "Accept",
61         "value": "*/*",
62         "name": "accept",
63         "type": "text"
64     },
65     {
66         "key": "Accept-Encoding",
67         "value": "gzip, deflate, br",
68         "name": "accept-encoding",
69         "type": "text"
70     },
71     {
```



```
68         {
69             "key": "Connection",
70             "value": "keep-alive",
71             "name": "connection",
72             "type": "text"
73         },
74         {
75             "key": "Accept",
76             "value": "application/json",
77             "type": "text"
78         },
79         {
80             "key": "X-API-KEY",
81             "value":
82                 "3EAF10998498A80C7F303674641F3D6C",
83             "type": "text"
84         },
85     ],
86     "body": {
87         "mode": "raw",
88         "raw": "",
89         "options": {
90             "raw": {
91                 "language": "json"
92             }
93         }
94     },
95     "url": {
96         "raw":
97             "http://127.0.0.1/workspace-prj/upload/api/departments.json",
98         "protocol": "http",
99         "host": [
100             "127",
101             "0",
102             "0",
103             "1"
104         ],
105         "path": [
106             "workspace-prj",
107             "upload",
108             "api",
109             "departments.json"
110         ]
111     }
112 }
```

```
109         }
110     },
111     "response": []
112 },
113 {
114     "name": "Tickets List",
115     "protocolProfileBehavior": {
116         "disableBodyPruning": true
117     },
118     "request": {
119         "method": "GET",
120         "header": [
121             {
122                 "key": "Cache-Control",
123                 "value": "no-cache",
124                 "name": "cache-control",
125                 "type": "text"
126             },
127             {
128                 "key": "Postman-Token",
129                 "value": "<calculated when request is
130                     sent>",
131                 "name": "postman-token",
132                 "type": "text"
133             },
134             {
135                 "key": "Content-Type",
136                 "value": "application/json",
137                 "name": "content-type",
138                 "type": "text"
139             },
140             {
141                 "key": "Content-Length",
142                 "value": "<calculated when request is
143                     sent>",
144                 "name": "content-length",
145                 "type": "text"
146             },
147             {
148                 "key": "Host",
149                 "value": "<calculated when request is
150                     sent>",
151                 "name": "host",
```

```
149         "type": "text"
150     },
151     {
152         "key": "User-Agent",
153         "value": "PostmanRuntime/7.39.1",
154         "name": "user-agent",
155         "type": "text"
156     },
157     {
158         "key": "Accept",
159         "value": "*/*",
160         "name": "accept",
161         "type": "text"
162     },
163     {
164         "key": "Accept-Encoding",
165         "value": "gzip, deflate, br",
166         "name": "accept-encoding",
167         "type": "text"
168     },
169     {
170         "key": "Connection",
171         "value": "keep-alive",
172         "name": "connection",
173         "type": "text"
174     },
175     {
176         "key": "Accept",
177         "value": "application/json",
178         "type": "text"
179     },
180     {
181         "key": "X-API-KEY",
182         "value":
183             "3EAF10998498A80C7F303674641F3D6C",
184         "type": "text"
185     }
186 ],
187 "body": {
188     "mode": "raw",
189     "raw": "",
190     "options": {
191         "raw": {
```

```

191         "language": "json"
192     }
193 }
194 },
195 "url": {
196     "raw":
197         "http://127.0.0.1/workspace-prj/upload/api/ticketsLi
198     "protocol": "http",
199     "host": [
200         "127",
201         "0",
202         "0",
203         "1"
204     ],
205     "path": [
206         "workspace-prj",
207         "upload",
208         "api",
209         "ticketsList.json"
210     ]
211 },
212 "response": []
213 },
214 {
215     "name": "SLAs",
216     "protocolProfileBehavior": {
217         "disableBodyPruning": true
218     },
219     "request": {
220         "method": "GET",
221         "header": [
222             {
223                 "key": "Cache-Control",
224                 "value": "no-cache",
225                 "name": "cache-control",
226                 "type": "text"
227             },
228             {
229                 "key": "Postman-Token",
230                 "value": "<calculated when request is
231                     sent>",

```

```
232         "type": "text"
233     },
234     {
235         "key": "Content-Type",
236         "value": "application/json",
237         "name": "content-type",
238         "type": "text"
239     },
240     {
241         "key": "Content-Length",
242         "value": "<calculated when request is
                sent>",
243         "name": "content-length",
244         "type": "text"
245     },
246     {
247         "key": "Host",
248         "value": "<calculated when request is
                sent>",
249         "name": "host",
250         "type": "text"
251     },
252     {
253         "key": "User-Agent",
254         "value": "PostmanRuntime/7.39.1",
255         "name": "user-agent",
256         "type": "text"
257     },
258     {
259         "key": "Accept",
260         "value": "*/*",
261         "name": "accept",
262         "type": "text"
263     },
264     {
265         "key": "Accept-Encoding",
266         "value": "gzip, deflate, br",
267         "name": "accept-encoding",
268         "type": "text"
269     },
270     {
271         "key": "Connection",
272         "value": "keep-alive",
```

```
273         "name": "connection",
274         "type": "text"
275     },
276     {
277         "key": "Accept",
278         "value": "application/json",
279         "type": "text"
280     },
281     {
282         "key": "X-API-KEY",
283         "value":
284             "3EAF10998498A80C7F303674641F3D6C",
285         "type": "text"
286     }
287 ],
288 "body": {
289     "mode": "raw",
290     "raw": "",
291     "options": {
292         "raw": {
293             "language": "json"
294         }
295     }
296 },
297 "url": {
298     "raw":
299         "http://127.0.0.1/workspace-prj/upload/api/slas.json",
300     "protocol": "http",
301     "host": [
302         "127",
303         "0",
304         "0",
305         "1"
306     ],
307     "path": [
308         "workspace-prj",
309         "upload",
310         "api",
311         "slas.json"
312     ]
313 },
314 "response": []
```

```
314     },
315     {
316         "name": "Teams",
317         "protocolProfileBehavior": {
318             "disableBodyPruning": true
319         },
320         "request": {
321             "method": "GET",
322             "header": [
323                 {
324                     "key": "Cache-Control",
325                     "value": "no-cache",
326                     "name": "cache-control",
327                     "type": "text"
328                 },
329                 {
330                     "key": "Postman-Token",
331                     "value": "<calculated when request is
332                         sent>",
333                     "name": "postman-token",
334                     "type": "text"
335                 },
336                 {
337                     "key": "Content-Type",
338                     "value": "application/json",
339                     "name": "content-type",
340                     "type": "text"
341                 },
342                 {
343                     "key": "Content-Length",
344                     "value": "<calculated when request is
345                         sent>",
346                     "name": "content-length",
347                     "type": "text"
348                 },
349                 {
350                     "key": "Host",
351                     "value": "<calculated when request is
352                         sent>",
353                     "name": "host",
354                     "type": "text"
355                 },
356                 {
```

```
354         "key": "User-Agent",
355         "value": "PostmanRuntime/7.39.1",
356         "name": "user-agent",
357         "type": "text"
358     },
359     {
360         "key": "Accept",
361         "value": "*/*",
362         "name": "accept",
363         "type": "text"
364     },
365     {
366         "key": "Accept-Encoding",
367         "value": "gzip, deflate, br",
368         "name": "accept-encoding",
369         "type": "text"
370     },
371     {
372         "key": "Connection",
373         "value": "keep-alive",
374         "name": "connection",
375         "type": "text"
376     },
377     {
378         "key": "Accept",
379         "value": "application/json",
380         "type": "text"
381     },
382     {
383         "key": "X-API-KEY",
384         "value":
385             "3EAF10998498A80C7F303674641F3D6C",
386         "type": "text"
387     },
388 ],
389 "body": {
390     "mode": "raw",
391     "raw": "",
392     "options": {
393         "raw": {
394             "language": "json"
395         }
396     }
397 }
```



```
396     },
397     "url": {
398         "raw":
399             "http://127.0.0.1/workspace-prj/upload/api/teams.json",
400         "protocol": "http",
401         "host": [
402             "127",
403             "0",
404             "0",
405             "1"
406         ],
407         "path": [
408             "workspace-prj",
409             "upload",
410             "api",
411             "teams.json"
412         ]
413     },
414     "response": []
415 },
416 {
417     "name": "Staff",
418     "protocolProfileBehavior": {
419         "disableBodyPruning": true
420     },
421     "request": {
422         "method": "GET",
423         "header": [
424             {
425                 "key": "Cache-Control",
426                 "value": "no-cache",
427                 "name": "cache-control",
428                 "type": "text"
429             },
430             {
431                 "key": "Postman-Token",
432                 "value": "<calculated when request is sent>",
433                 "name": "postman-token",
434                 "type": "text"
435             },
436             {
```

```
437         "key": "Content-Type",
438         "value": "application/json",
439         "name": "content-type",
440         "type": "text"
441     },
442     {
443         "key": "Content-Length",
444         "value": "<calculated when request is
            sent>",
445         "name": "content-length",
446         "type": "text"
447     },
448     {
449         "key": "Host",
450         "value": "<calculated when request is
            sent>",
451         "name": "host",
452         "type": "text"
453     },
454     {
455         "key": "User-Agent",
456         "value": "PostmanRuntime/7.39.1",
457         "name": "user-agent",
458         "type": "text"
459     },
460     {
461         "key": "Accept",
462         "value": "*/*",
463         "name": "accept",
464         "type": "text"
465     },
466     {
467         "key": "Accept-Encoding",
468         "value": "gzip, deflate, br",
469         "name": "accept-encoding",
470         "type": "text"
471     },
472     {
473         "key": "Connection",
474         "value": "keep-alive",
475         "name": "connection",
476         "type": "text"
477     },
```

```
478         {
479             "key": "Accept",
480             "value": "application/json",
481             "type": "text"
482         },
483         {
484             "key": "X-API-KEY",
485             "value":
486                 "3EAF10998498A80C7F303674641F3D6C",
487             "type": "text"
488         }
489     ],
490     "body": {
491         "mode": "raw",
492         "raw": "",
493         "options": {
494             "raw": {
495                 "language": "json"
496             }
497         }
498     },
499     "url": {
500         "raw":
501             "http://127.0.0.1/workspace-prj/upload/api/staff.json",
502         "protocol": "http",
503         "host": [
504             "127",
505             "0",
506             "0",
507             "1"
508         ],
509         "path": [
510             "workspace-prj",
511             "upload",
512             "api",
513             "staff.json"
514         ]
515     },
516     "response": []
517 },
518 {
519     "name": "Users",
```

```
519     "protocolProfileBehavior": {
520       "disableBodyPruning": true
521     },
522     "request": {
523       "method": "GET",
524       "header": [
525         {
526           "key": "Cache-Control",
527           "value": "no-cache",
528           "name": "cache-control",
529           "type": "text"
530         },
531         {
532           "key": "Postman-Token",
533           "value": "<calculated when request is
                    sent>",
534           "name": "postman-token",
535           "type": "text"
536         },
537         {
538           "key": "Content-Type",
539           "value": "application/json",
540           "name": "content-type",
541           "type": "text"
542         },
543         {
544           "key": "Content-Length",
545           "value": "<calculated when request is
                    sent>",
546           "name": "content-length",
547           "type": "text"
548         },
549         {
550           "key": "Host",
551           "value": "<calculated when request is
                    sent>",
552           "name": "host",
553           "type": "text"
554         },
555         {
556           "key": "User-Agent",
557           "value": "PostmanRuntime/7.39.1",
558           "name": "user-agent",
```

```
559         "type": "text"
560     },
561     {
562         "key": "Accept",
563         "value": "*/*",
564         "name": "accept",
565         "type": "text"
566     },
567     {
568         "key": "Accept-Encoding",
569         "value": "gzip, deflate, br",
570         "name": "accept-encoding",
571         "type": "text"
572     },
573     {
574         "key": "Connection",
575         "value": "keep-alive",
576         "name": "connection",
577         "type": "text"
578     },
579     {
580         "key": "Accept",
581         "value": "application/json",
582         "type": "text"
583     },
584     {
585         "key": "X-API-KEY",
586         "value":
587             "3EAF10998498A80C7F303674641F3D6C",
588         "type": "text"
589     },
590     ],
591     "body": {
592         "mode": "raw",
593         "raw": "",
594         "options": {
595             "raw": {
596                 "language": "json"
597             }
598         }
599     },
600     "url": {
601         "raw":
```

```
        "http://127.0.0.1/workspace-prj/upload/api/users.json",
601      "protocol": "http",
602      "host": [
603        "127",
604        "0",
605        "0",
606        "1"
607      ],
608      "path": [
609        "workspace-prj",
610        "upload",
611        "api",
612        "users.json"
613      ]
614    }
615  },
616  "response": []
617},
618{
619  "name": "Priorities",
620  "protocolProfileBehavior": {
621    "disableBodyPruning": true
622  },
623  "request": {
624    "method": "GET",
625    "header": [
626      {
627        "key": "Cache-Control",
628        "value": "no-cache",
629        "name": "cache-control",
630        "type": "text"
631      },
632      {
633        "key": "Postman-Token",
634        "value": "<calculated when request is sent>",
635        "name": "postman-token",
636        "type": "text"
637      },
638      {
639        "key": "Content-Type",
640        "value": "application/json",
641        "name": "content-type",
```

```
642         "type": "text"
643     },
644     {
645         "key": "Content-Length",
646         "value": "<calculated when request is
        sent>",
647         "name": "content-length",
648         "type": "text"
649     },
650     {
651         "key": "Host",
652         "value": "<calculated when request is
        sent>",
653         "name": "host",
654         "type": "text"
655     },
656     {
657         "key": "User-Agent",
658         "value": "PostmanRuntime/7.39.1",
659         "name": "user-agent",
660         "type": "text"
661     },
662     {
663         "key": "Accept",
664         "value": "*/*",
665         "name": "accept",
666         "type": "text"
667     },
668     {
669         "key": "Accept-Encoding",
670         "value": "gzip, deflate, br",
671         "name": "accept-encoding",
672         "type": "text"
673     },
674     {
675         "key": "Connection",
676         "value": "keep-alive",
677         "name": "connection",
678         "type": "text"
679     },
680     {
681         "key": "Accept",
682         "value": "application/json",
```

```

683         "type": "text"
684     },
685     {
686         "key": "X-API-KEY",
687         "value":
688             "3EAF10998498A80C7F303674641F3D6C",
689         "type": "text"
690     }
691 ],
692 "body": {
693     "mode": "raw",
694     "raw": "",
695     "options": {
696         "raw": {
697             "language": "json"
698         }
699     },
700     "url": {
701         "raw":
702             "http://127.0.0.1/workspace-prj/upload/api/priorities",
703         "protocol": "http",
704         "host": [
705             "127",
706             "0",
707             "0",
708             "1"
709         ],
710         "path": [
711             "workspace-prj",
712             "upload",
713             "api",
714             "priorities.json"
715         ]
716     },
717     "response": []
718 },
719 {
720     "name": "Topics",
721     "protocolProfileBehavior": {
722         "disableBodyPruning": true
723     },

```



```
724     "request": {
725         "method": "GET",
726         "header": [
727             {
728                 "key": "Cache-Control",
729                 "value": "no-cache",
730                 "name": "cache-control",
731                 "type": "text"
732             },
733             {
734                 "key": "Postman-Token",
735                 "value": "<calculated when request is
736                     sent>",
737                 "name": "postman-token",
738                 "type": "text"
739             },
740             {
741                 "key": "Content-Type",
742                 "value": "application/json",
743                 "name": "content-type",
744                 "type": "text"
745             },
746             {
747                 "key": "Content-Length",
748                 "value": "<calculated when request is
749                     sent>",
750                 "name": "content-length",
751                 "type": "text"
752             },
753             {
754                 "key": "Host",
755                 "value": "<calculated when request is
756                     sent>",
757                 "name": "host",
758                 "type": "text"
759             },
760             {
761                 "key": "User-Agent",
762                 "value": "PostmanRuntime/7.39.1",
763                 "name": "user-agent",
764                 "type": "text"
765             },
766             {
```

```

764         "key": "Accept",
765         "value": "*/*",
766         "name": "accept",
767         "type": "text"
768     },
769     {
770         "key": "Accept-Encoding",
771         "value": "gzip, deflate, br",
772         "name": "accept-encoding",
773         "type": "text"
774     },
775     {
776         "key": "Connection",
777         "value": "keep-alive",
778         "name": "connection",
779         "type": "text"
780     },
781     {
782         "key": "Accept",
783         "value": "application/json",
784         "type": "text"
785     },
786     {
787         "key": "X-API-KEY",
788         "value":
789             "3EAF10998498A80C7F303674641F3D6C",
790         "type": "text"
791     }
792 ],
793 "body": {
794     "mode": "raw",
795     "raw": "",
796     "options": {
797         "raw": {
798             "language": "json"
799         }
800     }
801 },
802 "url": {
803     "raw":
804         "http://127.0.0.1/workspace-prj/upload/api/topics.js
805     "protocol": "http",
806     "host": [

```

```
805         "127",
806         "0",
807         "0",
808         "1"
809     ],
810     "path": [
811         "workspace-prj",
812         "upload",
813         "api",
814         "topics.json"
815     ]
816 },
817 },
818 "response": []
819 },
820 {
821     "name": "Sources",
822     "protocolProfileBehavior": {
823         "disableBodyPruning": true
824     },
825     "request": {
826         "method": "GET",
827         "header": [
828             {
829                 "key": "Cache-Control",
830                 "value": "no-cache",
831                 "name": "cache-control",
832                 "type": "text"
833             },
834             {
835                 "key": "Postman-Token",
836                 "value": "<calculated when request is  
sent>",
837                 "name": "postman-token",
838                 "type": "text"
839             },
840             {
841                 "key": "Content-Type",
842                 "value": "application/json",
843                 "name": "content-type",
844                 "type": "text"
845             },
846             {
```

```
847         "key": "Content-Length",
848         "value": "<calculated when request is
            sent>",
849         "name": "content-length",
850         "type": "text"
851     },
852     {
853         "key": "Host",
854         "value": "<calculated when request is
            sent>",
855         "name": "host",
856         "type": "text"
857     },
858     {
859         "key": "User-Agent",
860         "value": "PostmanRuntime/7.39.1",
861         "name": "user-agent",
862         "type": "text"
863     },
864     {
865         "key": "Accept",
866         "value": "*/*",
867         "name": "accept",
868         "type": "text"
869     },
870     {
871         "key": "Accept-Encoding",
872         "value": "gzip, deflate, br",
873         "name": "accept-encoding",
874         "type": "text"
875     },
876     {
877         "key": "Connection",
878         "value": "keep-alive",
879         "name": "connection",
880         "type": "text"
881     },
882     {
883         "key": "Accept",
884         "value": "application/json",
885         "type": "text"
886     },
887     {
```

```
888         "key": "X-API-KEY",
889         "value":
890             "3EAF10998498A80C7F303674641F3D6C",
891         "type": "text"
892     },
893     "body": {
894         "mode": "raw",
895         "raw": "",
896         "options": {
897             "raw": {
898                 "language": "json"
899             }
900         }
901     },
902     "url": {
903         "raw":
904             "http://127.0.0.1/workspace-prj/upload/api/sources.json",
905         "protocol": "http",
906         "host": [
907             "127",
908             "0",
909             "0",
910             "1"
911         ],
912         "path": [
913             "workspace-prj",
914             "upload",
915             "api",
916             "sources.json"
917         ]
918     },
919     "response": []
920 }
921 ]
922 },
923 {
924     "name": "Posts",
925     "item": [
926         {
927             "name": "Suspend",
928             "request": {
```

```
929     "method": "POST",
930     "header": [
931         {
932             "key": "Cache-Control",
933             "value": "no-cache",
934             "name": "cache-control",
935             "type": "text"
936         },
937         {
938             "key": "Postman-Token",
939             "value": "<calculated when request is
940                 sent>",
941             "name": "postman-token",
942             "type": "text"
943         },
944         {
945             "key": "Content-Type",
946             "value": "application/json",
947             "name": "content-type",
948             "type": "text"
949         },
950         {
951             "key": "Content-Length",
952             "value": "<calculated when request is
953                 sent>",
954             "name": "content-length",
955             "type": "text"
956         },
957         {
958             "key": "Host",
959             "value": "<calculated when request is
960                 sent>",
961             "name": "host",
962             "type": "text"
963         },
964         {
965             "key": "User-Agent",
966             "value": "PostmanRuntime/7.39.1",
967             "name": "user-agent",
968             "type": "text"
969         },
970         {
971             "key": "Accept",
```

```

969         "value": "/*/*",
970         "name": "accept",
971         "type": "text"
972     },
973     {
974         "key": "Accept-Encoding",
975         "value": "gzip, deflate, br",
976         "name": "accept-encoding",
977         "type": "text"
978     },
979     {
980         "key": "Connection",
981         "value": "keep-alive",
982         "name": "connection",
983         "type": "text"
984     },
985     {
986         "key": "Accept",
987         "value": "application/json",
988         "type": "text"
989     },
990     {
991         "key": "X-API-KEY",
992         "value":
993             "7DDFB6CCCC235070F7A0B1A0955992BB",
994         "type": "text"
995     }
996 ],
997 "body": {
998     "mode": "raw",
999     "raw": "{\r\n    \"ticketNumber\":\r\n        \"644099\", \r\n    \"comments\":\r\n        \"Default comments\"\r\n}",
1000     "options": {
1001         "raw": {
1002             "language": "json"
1003         }
1004     },
1005     "url": {
1006         "raw":
1007             "http://127.0.0.1/workspace-prj/upload/api/suspend/tickets
1008         "protocol": "http",

```

```
1008         "host": [  
1009             "127",  
1010             "0",  
1011             "0",  
1012             "1"  
1013         ],  
1014         "path": [  
1015             "workspace-prj",  
1016             "upload",  
1017             "api",  
1018             "suspend",  
1019             "tickets.json"  
1020         ]  
1021     }  
1022 },  
1023     "response": []  
1024 },  
1025 {  
1026     "name": "Delete",  
1027     "request": {  
1028         "method": "POST",  
1029         "header": [  
1030             {  
1031                 "key": "Cache-Control",  
1032                 "value": "no-cache",  
1033                 "name": "cache-control",  
1034                 "type": "text"  
1035             },  
1036             {  
1037                 "key": "Postman-Token",  
1038                 "value": "<calculated when request is  
1039                     sent>",  
1040                 "name": "postman-token",  
1041                 "type": "text"  
1042             },  
1043             {  
1044                 "key": "Content-Type",  
1045                 "value": "application/json",  
1046                 "name": "content-type",  
1047                 "type": "text"  
1048             },  
1049             {  
1050                 "key": "Content-Length",
```



```
1050         "value": "<calculated when request is
1051             sent>",
1052         "name": "content-length",
1053         "type": "text"
1054     },
1055     {
1056         "key": "Host",
1057         "value": "<calculated when request is
1058             sent>",
1059         "name": "host",
1060         "type": "text"
1061     },
1062     {
1063         "key": "User-Agent",
1064         "value": "PostmanRuntime/7.39.1",
1065         "name": "user-agent",
1066         "type": "text"
1067     },
1068     {
1069         "key": "Accept",
1070         "value": "*/*",
1071         "name": "accept",
1072         "type": "text"
1073     },
1074     {
1075         "key": "Accept-Encoding",
1076         "value": "gzip, deflate, br",
1077         "name": "accept-encoding",
1078         "type": "text"
1079     },
1080     {
1081         "key": "Connection",
1082         "value": "keep-alive",
1083         "name": "connection",
1084         "type": "text"
1085     },
1086     {
1087         "key": "Accept",
1088         "value": "application/json",
1089         "type": "text"
1090     },
1091     {
1092         "key": "X-API-KEY",
```

```

1091         "value":
1092             "7DDFB6CCCC235070F7A0B1A0955992BB",
1093         "type": "text"
1094     },
1095     "body": {
1096         "mode": "raw",
1097         "raw": "{\r\n    \"ticketNumber\":\r\n    \"882901\", \r\n    \"comments\":\r\n    \"Default comments\"\r\n}",
1098         "options": {
1099             "raw": {
1100                 "language": "json"
1101             }
1102         }
1103     },
1104     "url": {
1105         "raw":
1106             "http://127.0.0.1/workspace-prj/upload/api/delete/ti
1107         "protocol": "http",
1108         "host": [
1109             "127",
1110             "0",
1111             "0",
1112             "1",
1113         ],
1114         "path": [
1115             "workspace-prj",
1116             "upload",
1117             "api",
1118             "delete",
1119             "tickets.json"
1120         ]
1121     },
1122     "response": []
1123 },
1124 {
1125     "name": "GetAPIKey",
1126     "request": {
1127         "method": "POST",
1128         "header": [
1129             {

```

```
1130         "key": "Cache-Control",
1131         "value": "no-cache",
1132         "name": "cache-control",
1133         "type": "text"
1134     },
1135     {
1136         "key": "Postman-Token",
1137         "value": "<calculated when request is
            sent>",
1138         "name": "postman-token",
1139         "type": "text"
1140     },
1141     {
1142         "key": "Content-Type",
1143         "value": "application/json",
1144         "name": "content-type",
1145         "type": "text"
1146     },
1147     {
1148         "key": "Content-Length",
1149         "value": "<calculated when request is
            sent>",
1150         "name": "content-length",
1151         "type": "text"
1152     },
1153     {
1154         "key": "Host",
1155         "value": "<calculated when request is
            sent>",
1156         "name": "host",
1157         "type": "text"
1158     },
1159     {
1160         "key": "User-Agent",
1161         "value": "PostmanRuntime/7.39.1",
1162         "name": "user-agent",
1163         "type": "text"
1164     },
1165     {
1166         "key": "Accept",
1167         "value": "/*/*",
1168         "name": "accept",
1169         "type": "text"
```

```

1170         },
1171         {
1172             "key": "Accept-Encoding",
1173             "value": "gzip, deflate, br",
1174             "name": "accept-encoding",
1175             "type": "text"
1176         },
1177         {
1178             "key": "Connection",
1179             "value": "keep-alive",
1180             "name": "connection",
1181             "type": "text"
1182         },
1183         {
1184             "key": "Accept",
1185             "value": "application/json",
1186             "type": "text"
1187         },
1188         {
1189             "key": "X-API-KEY",
1190             "value":
1191                 "7DDFB6CCCC235070F7A0B1A0955992BB",
1192             "type": "text"
1193     ],
1194     "body": {
1195         "mode": "raw",
1196         "raw": "{\r\n
1197             \"staff\": \"antFerreira\", \r\n
1198             \"password\": \"123123\" \r\n}",
1199         "options": {
1200             "raw": {
1201                 "language": "json"
1202             }
1203         }
1204     },
1205     "url": {
1206         "raw":
1207             "http://127.0.0.1/workspace-prj/upload/api/getApiKey",
1208         "protocol": "http",
1209         "host": [
1210             "127",
1211             "0",

```

```
1209         "0",
1210         "1"
1211     ],
1212     "path": [
1213         "workspace-prj",
1214         "upload",
1215         "api",
1216         "getApiKey",
1217         "tickets.json"
1218     ]
1219 },
1220 },
1221 "response": []
1222 },
1223 {
1224     "name": "unSuspend",
1225     "request": {
1226         "method": "POST",
1227         "header": [
1228             {
1229                 "key": "Cache-Control",
1230                 "value": "no-cache",
1231                 "name": "cache-control",
1232                 "type": "text"
1233             },
1234             {
1235                 "key": "Postman-Token",
1236                 "value": "<calculated when request is
1237                     sent>",
1238                 "name": "postman-token",
1239                 "type": "text"
1240             },
1241             {
1242                 "key": "Content-Type",
1243                 "value": "application/json",
1244                 "name": "content-type",
1245                 "type": "text"
1246             },
1247             {
1248                 "key": "Content-Length",
1249                 "value": "<calculated when request is
1250                     sent>",
1251                 "name": "content-length",
```

```
1250         "type": "text"
1251     },
1252     {
1253         "key": "Host",
1254         "value": "<calculated when request is
            sent>",
1255         "name": "host",
1256         "type": "text"
1257     },
1258     {
1259         "key": "User-Agent",
1260         "value": "PostmanRuntime/7.39.1",
1261         "name": "user-agent",
1262         "type": "text"
1263     },
1264     {
1265         "key": "Accept",
1266         "value": "*/*",
1267         "name": "accept",
1268         "type": "text"
1269     },
1270     {
1271         "key": "Accept-Encoding",
1272         "value": "gzip, deflate, br",
1273         "name": "accept-encoding",
1274         "type": "text"
1275     },
1276     {
1277         "key": "Connection",
1278         "value": "keep-alive",
1279         "name": "connection",
1280         "type": "text"
1281     },
1282     {
1283         "key": "Accept",
1284         "value": "application/json",
1285         "type": "text"
1286     },
1287     {
1288         "key": "X-API-KEY",
1289         "value":
1290             "3EAF10998498A80C7F303674641F3D6C",
1291         "type": "text"
```

```
1291     }
1292 ],
1293 "body": {
1294     "mode": "raw",
1295     "raw": "{\r\n        \"ticketNumber\":
1296         \"424628\", \r\n        \"comments\":
1297         \"Default comments\"\r\n}",
1298     "options": {
1299         "raw": {
1300             "language": "json"
1301         }
1302     },
1303     "url": {
1304         "raw":
1305             "http://127.0.0.1/workspace-prj/upload/api/unSuspend/ticke
1306         "protocol": "http",
1307         "host": [
1308             "127",
1309             "0",
1310             "0",
1311             "1"
1312         ],
1313         "path": [
1314             "workspace-prj",
1315             "upload",
1316             "api",
1317             "unSuspend",
1318             "tickets.json"
1319         ]
1320     },
1321     "response": []
1322 },
1323 {
1324     "name": "RequestApiKey",
1325     "request": {
1326         "method": "POST",
1327         "header": [
1328             {
1329                 "key": "Cache-Control",
1330                 "value": "no-cache",
1331                 "name": "cache-control",
```

```
1331         "type": "text"
1332     },
1333     {
1334         "key": "Postman-Token",
1335         "value": "<calculated when request is
1336             sent>",
1337         "name": "postman-token",
1338         "type": "text"
1339     },
1340     {
1341         "key": "Content-Type",
1342         "value": "application/json",
1343         "name": "content-type",
1344         "type": "text"
1345     },
1346     {
1347         "key": "Content-Length",
1348         "value": "<calculated when request is
1349             sent>",
1350         "name": "content-length",
1351         "type": "text"
1352     },
1353     {
1354         "key": "Host",
1355         "value": "<calculated when request is
1356             sent>",
1357         "name": "host",
1358         "type": "text"
1359     },
1360     {
1361         "key": "User-Agent",
1362         "value": "PostmanRuntime/7.39.1",
1363         "name": "user-agent",
1364         "type": "text"
1365     },
1366     {
1367         "key": "Accept",
1368         "value": "*/*",
1369         "name": "accept",
1370         "type": "text"
```

```

1371         "value": "gzip, deflate, br",
1372         "name": "accept-encoding",
1373         "type": "text"
1374     },
1375     {
1376         "key": "Connection",
1377         "value": "keep-alive",
1378         "name": "connection",
1379         "type": "text"
1380     },
1381     {
1382         "key": "Accept",
1383         "value": "application/json",
1384         "type": "text"
1385     }
1386 ],
1387 "body": {
1388     "mode": "raw",
1389     "raw": "{\r\n
1390         \"admin\": \"antFerreira\", \r\n
1391         \"adminPassword\": \"123123\", \r\n
1392         \"staff\": \"antFerreira\", \r\n
1393         \"isActive\": \"1\", \r\n
1394         \"canCreateTickets\": \"1\", \r\n
1395         \"canCloseTickets\": \"1\", \r\n
1396         \"canReopenTickets\": \"1\", \r\n
1397         \"canEditTickets\": \"1\", \r\n
1398         \"canSuspendTickets\": \"1\", \r\n
1399         \"notes\": \"notas do admin\" \r\n}",
1400     "options": {
1401         "raw": {
1402             "language": "json"

```

```
1403         "1"
1404     ],
1405     "path": [
1406         "workspace-prj",
1407         "upload",
1408         "api",
1409         "requestApiKey",
1410         "tickets.json"
1411     ]
1412 }
1413 },
1414 "response": []
1415 },
1416 {
1417     "name": "Open",
1418     "request": {
1419         "method": "POST",
1420         "header": [
1421             {
1422                 "key": "Cache-Control",
1423                 "value": "no-cache",
1424                 "name": "cache-control",
1425                 "type": "text"
1426             },
1427             {
1428                 "key": "Postman-Token",
1429                 "value": "<calculated when request is
1430                     sent>",
1431                 "name": "postman-token",
1432                 "type": "text"
1433             },
1434             {
1435                 "key": "Content-Type",
1436                 "value": "application/json",
1437                 "name": "content-type",
1438                 "type": "text"
1439             },
1440             {
1441                 "key": "Content-Length",
1442                 "value": "<calculated when request is
1443                     sent>",
1444                 "name": "content-length",
1445                 "type": "text"
```

```
1444     },
1445     {
1446         "key": "Host",
1447         "value": "<calculated when request is
            sent>",
1448         "name": "host",
1449         "type": "text"
1450     },
1451     {
1452         "key": "User-Agent",
1453         "value": "PostmanRuntime/7.39.1",
1454         "name": "user-agent",
1455         "type": "text"
1456     },
1457     {
1458         "key": "Accept",
1459         "value": "*/*",
1460         "name": "accept",
1461         "type": "text"
1462     },
1463     {
1464         "key": "Accept-Encoding",
1465         "value": "gzip, deflate, br",
1466         "name": "accept-encoding",
1467         "type": "text"
1468     },
1469     {
1470         "key": "Connection",
1471         "value": "keep-alive",
1472         "name": "connection",
1473         "type": "text"
1474     },
1475     {
1476         "key": "Accept",
1477         "value": "application/json",
1478         "type": "text"
1479     },
1480     {
1481         "key": "X-API-KEY",
1482         "value":
            "7DDFB6CCCC235070F7A0B1A0955992BB",
1483         "type": "text"
1484     }
```

```

1485     ],
1486     "body": {
1487         "mode": "raw",
1488         "raw": "{\r\n        \"subject\":
            \"Subject\", \r\n
            \"message\": \"Default message\", \r\n
            \"attachments\":
            [{\"attachment.txt\":
            \"data:text/plain; charset=utf-8, content
            of attachment\"}], \r\n
            \"topicId\": \"1\" \r\n} \r\n \r\n",
1489     "options": {
1490         "raw": {
1491             "language": "json"
1492         }
1493     }
1494 },
1495 "url": {
1496     "raw":
1497         "http://127.0.0.1/workspace-prj/upload/api/open/tick
1498     "protocol": "http",
1499     "host": [
1500         "127",
1501         "0",
1502         "0",
1503         "1",
1504     ],
1505     "path": [
1506         "workspace-prj",
1507         "upload",
1508         "api",
1509         "open",
1510         "tickets.json"
1511     ]
1512 },
1513 "response": []
1514 },
1515 {
1516     "name": "Close",
1517     "request": {
1518         "method": "POST",
1519         "header": [

```

```
1520     {
1521         "key": "Cache-Control",
1522         "value": "no-cache",
1523         "name": "cache-control",
1524         "type": "text"
1525     },
1526     {
1527         "key": "Postman-Token",
1528         "value": "<calculated when request is
1529             sent>",
1529         "name": "postman-token",
1530         "type": "text"
1531     },
1532     {
1533         "key": "Content-Type",
1534         "value": "application/json",
1535         "name": "content-type",
1536         "type": "text"
1537     },
1538     {
1539         "key": "Content-Length",
1540         "value": "<calculated when request is
1541             sent>",
1541         "name": "content-length",
1542         "type": "text"
1543     },
1544     {
1545         "key": "Host",
1546         "value": "<calculated when request is
1547             sent>",
1547         "name": "host",
1548         "type": "text"
1549     },
1550     {
1551         "key": "User-Agent",
1552         "value": "PostmanRuntime/7.39.1",
1553         "name": "user-agent",
1554         "type": "text"
1555     },
1556     {
1557         "key": "Accept",
1558         "value": "*/*",
1559         "name": "accept",
```

```

1560         "type": "text"
1561     },
1562     {
1563         "key": "Accept-Encoding",
1564         "value": "gzip, deflate, br",
1565         "name": "accept-encoding",
1566         "type": "text"
1567     },
1568     {
1569         "key": "Connection",
1570         "value": "keep-alive",
1571         "name": "connection",
1572         "type": "text"
1573     },
1574     {
1575         "key": "Accept",
1576         "value": "application/json",
1577         "type": "text"
1578     },
1579     {
1580         "key": "X-API-KEY",
1581         "value":
1582             "7DDFB6CCCC235070F7A0B1A0955992BB",
1583         "type": "text"
1584     },
1585     ],
1586     "body": {
1587         "mode": "raw",
1588         "raw": "{\r\n    \"ticketNumber\":\r\n    \"644099\", \r\n    \"comments\":\r\n    \"Default comments\"\r\n}\r\n",
1589         "options": {
1590             "raw": {
1591                 "language": "json"
1592             }
1593         },
1594         "url": {
1595             "raw":
1596                 "http://127.0.0.1/workspace-prj/upload/api/close/tic
1597             "protocol": "http",
1598             "host": [
1599                 "127",

```

```
1599         "0",
1600         "0",
1601         "1"
1602     ],
1603     "path": [
1604         "workspace-prj",
1605         "upload",
1606         "api",
1607         "close",
1608         "tickets.json"
1609     ]
1610 }
1611 },
1612 "response": []
1613 },
1614 {
1615     "name": "ReOpen",
1616     "request": {
1617         "method": "POST",
1618         "header": [
1619             {
1620                 "key": "Cache-Control",
1621                 "value": "no-cache",
1622                 "name": "cache-control",
1623                 "type": "text"
1624             },
1625             {
1626                 "key": "Postman-Token",
1627                 "value": "<calculated when request is
1628                     sent>",
1629                 "name": "postman-token",
1630                 "type": "text"
1631             },
1632             {
1633                 "key": "Content-Type",
1634                 "value": "application/json",
1635                 "name": "content-type",
1636                 "type": "text"
1637             },
1638             {
1639                 "key": "Content-Length",
1640                 "value": "<calculated when request is
1641                     sent>",
```

```
1640         "name": "content-length",
1641         "type": "text"
1642     },
1643     {
1644         "key": "Host",
1645         "value": "<calculated when request is
1646             sent>",
1647         "name": "host",
1648         "type": "text"
1649     },
1650     {
1651         "key": "User-Agent",
1652         "value": "PostmanRuntime/7.39.1",
1653         "name": "user-agent",
1654         "type": "text"
1655     },
1656     {
1657         "key": "Accept",
1658         "value": "*/*",
1659         "name": "accept",
1660         "type": "text"
1661     },
1662     {
1663         "key": "Accept-Encoding",
1664         "value": "gzip, deflate, br",
1665         "name": "accept-encoding",
1666         "type": "text"
1667     },
1668     {
1669         "key": "Connection",
1670         "value": "keep-alive",
1671         "name": "connection",
1672         "type": "text"
1673     },
1674     {
1675         "key": "Accept",
1676         "value": "application/json",
1677         "type": "text"
1678     },
1679     {
1680         "key": "X-API-KEY",
1681         "value":
1682             "B7C78B841E4B9E2265CB4674AF6E68A7",
```



```
1681         "type": "text"
1682     }
1683 ],
1684 "body": {
1685     "mode": "raw",
1686     "raw": "{\r\n    \"ticketNumber\":\n\n    \"644099\", \r\n    \"comments\":\n\n    \"Default comments\"\r\n}\r\n",
1687     "options": {
1688         "raw": {
1689             "language": "json"
1690         }
1691     }
1692 },
1693 "url": {
1694     "raw":
1695         "http://127.0.0.1/workspace-prj/upload/api/reopen/tickets.",
1696     "protocol": "http",
1697     "host": [
1698         "127",
1699         "0",
1700         "0",
1701         "1"
1702     ],
1703     "path": [
1704         "workspace-prj",
1705         "upload",
1706         "api",
1707         "reopen",
1708         "tickets.json"
1709     ]
1710 },
1711 "response": []
1712 },
1713 {
1714     "name": "Edit",
1715     "request": {
1716         "method": "POST",
1717         "header": [
1718             {
1719                 "key": "Cache-Control",
1720                 "value": "no-cache",
```

```
1721         "name": "cache-control",
1722         "type": "text"
1723     },
1724     {
1725         "key": "Postman-Token",
1726         "value": "<calculated when request is
            sent>",
1727         "name": "postman-token",
1728         "type": "text"
1729     },
1730     {
1731         "key": "Content-Type",
1732         "value": "application/json",
1733         "name": "content-type",
1734         "type": "text"
1735     },
1736     {
1737         "key": "Content-Length",
1738         "value": "<calculated when request is
            sent>",
1739         "name": "content-length",
1740         "type": "text"
1741     },
1742     {
1743         "key": "Host",
1744         "value": "<calculated when request is
            sent>",
1745         "name": "host",
1746         "type": "text"
1747     },
1748     {
1749         "key": "User-Agent",
1750         "value": "PostmanRuntime/7.39.1",
1751         "name": "user-agent",
1752         "type": "text"
1753     },
1754     {
1755         "key": "Accept",
1756         "value": "*/*",
1757         "name": "accept",
1758         "type": "text"
1759     },
1760     {
```

```

1761         "key": "Accept-Encoding",
1762         "value": "gzip, deflate, br",
1763         "name": "accept-encoding",
1764         "type": "text"
1765     },
1766     {
1767         "key": "Connection",
1768         "value": "keep-alive",
1769         "name": "connection",
1770         "type": "text"
1771     },
1772     {
1773         "key": "Accept",
1774         "value": "application/json",
1775         "type": "text"
1776     },
1777     {
1778         "key": "X-API-KEY",
1779         "value":
1780             "3EAF10998498A80C7F303674641F3D6C",
1781         "type": "text"
1782     }
1783 ],
1784 "body": {
1785     "mode": "raw",
1786     "raw": "{\r\n    \"ticketNumber\":\r\n        \"293046\", \r\n    \"comments\":\r\n        \"Default comments\", \r\n    \"staff\": \"1\", \r\n    \"team\":\r\n        \"2\", \r\n    \"duedate\":\r\n        \"2024-07-20 20:00:00\", \r\n    \"dept\": \"2\", \r\n    \"refer\":\r\n        \"true\", \r\n    \"sla\": \"2\", \r\n        \"priority\": \"1\", \r\n    \"topic\": \"2\", \r\n    \"user\":\r\n        \"3\" \r\n}\r\n",
1787     "options": {
1788         "raw": {
1789             "language": "json"
1790         }
1791     },
1792     "url": {

```

```
1793         "raw":
1794             "http://127.0.0.1/workspace-prj/upload/api/edit/tick
1795         "protocol": "http",
1796         "host": [
1797             "127",
1798             "0",
1799             "0",
1800             "1"
1801         ],
1802         "path": [
1803             "workspace-prj",
1804             "upload",
1805             "api",
1806             "edit",
1807             "tickets.json"
1808         ]
1809     },
1810     "response": []
1811 }
1812 ]
1813 }
1814 ]
1815 }
```

Bibliografia

[Grupo15, 2024] Grupo15 (2024). Projeto ansr isel. <https://tomasgomesisel.gitbook.io/projeto-ansr-isel/>.

[Grupo15_online_rep, 2024] Grupo15_online_rep (2024). Projeto ansr isel github. <https://github.com/pedrosilva22000/OSTicket-API-Extension>.