Group 40 Project Step 5 (Portfolio Assignment)
Team Name: Team Data Wranglers
Team Members: Zhen Jie Chong, Zhen Kang Chong

Service Request Management System

Website URL: http://classwork.engr.oregonstate.edu:24658/

1.1. Executive Summary

This section documents the major changes we made in each step of our project, affecting the overall database design and implementation.

Actions Based on Feedback

i. In Project Step 1, the key problem we had was the lack of context in our project and database outlines for the management of "service requests". As service requests could exist in different domains such as logistics, healthcare, and facilities management, this made our problem statement very generic. From there, we defined our service requests to be technology focused, looking into areas like cybersecurity and artificial intelligence.

ii. In Project Step 2, we changed the Payments table to a standalone entity without any FKs and added the payment_id under the Requests entity as FK instead. We can think of the payment details as characteristics of each request made. Since payment can be optional (not yet paid), the value displayed will be NULL.

iii. In Project Step 3, the feedback was focused on the UI design. These will not be elaborated here as it was just a matter of time constraints for us during the draft.

iv. In Project Step 4, the main issues were failing to implement the .env file for reading credentials on the server (no issue faced locally) and intermittent lags when submitting the forms on the UI. The reading of .env files has been resolved (likely due to some errors we made unknowingly). For the intermittent lags, we tried changing the port number; the UI appears to work fine now, refreshing the page after each CRUD operation.

Upgrades from Previous Drafts

i. We removed service_price under Services and added quotation_price under RequestServices. Instead of tagging a fixed price to each service, this allows the price to be flexible in each request, depending on factors like the level of customization required.

ii. We used basic exception handling to direct users to an error page when trying to create or update a record that results in duplicates for certain columns following the schema.

iii. Following the example UI shared by James Coles (TA) on Ed Discussion (cited in our README.md), we thought it was a good idea to restrict the deletion of records on certain tables until the condition is met, which helps to improve data integrity on top of the constraints set in our schema. Besides this, we added a search filter on the Requested Services page to narrow down the results based on the specified Service, which would be of interest to identify trends and patterns related to that Service.

1.2. Project Outline

ProService Solutions is a well-established information technology (IT) service provider that prides itself on offering a wide range of 15 distinct Services, categorized into 3 different ServiceCategories: artificial intelligence, software development and cybersecurity. With more than 2,000 return Customers and 5,000 Requests handled annually, the company is planning to scale its business operations. The lack of a streamlined system may lead to lost Requests, delayed Services and frustrated Customers. As an IT company, it strives to stay at the forefront of technological advancements and keep up with industry standards. As such, the company is looking to adopt a database-driven website early to facilitate RequestServices tracking and process Payments smoothly. Overall, this helps to improve productivity and increase the satisfaction of Customers.

1.3. Database Outline

PK: Primary Key, FK: Foreign Key, NN: Not Null, UQ: Unique, AI: Auto-incremented

Customers

Records the details of Customers who interacted with ProService Solutions. The main details include contact information of the Customers for better communication.

- **Attributes:**
    - customer_id: INT, AI, UQ, NN, PK
    - first_name: VARCHAR(100), NN
    - last_name: VARCHAR(100), NN
    - address: VARCHAR(200), NN
    - phone: VARCHAR(20), UQ, NN
    - email: VARCHAR(50), UQ, NN
- **Relationship:**
    - An optional 1:M relationship between Customers and Requests is implemented with customer_id as a FK inside of Requests.
        - Each request must be associated with a customer, but each customer may have zero or many requests. The customer can be registered in the database but has yet to make a request.

Services

Records the details of Services offered by ProService Solutions. The main details include descriptive information of the Services provided and the distinct category they belong to.

- **Attributes:**
    - service_id: INT, AI, UQ, NN, PK
    - name: VARCHAR(100), UQ, NN
    - description: VARCHAR(200), NN
    - service_category_id: INT, FK
- **Relationship:**

    - An optional M:N relationship between Services and Requests is implemented with service_id and request_id as FKs inside of RequestServices.
        - Each request must consist of one or many services, but each service may be requested zero or many times. There can be a service which has never been requested before, an example being a service that is newly added.
    - A 1:M relationship between Services and RequestServices (intersection table).
    - A M:1 relationship between Services and ServiceCategories is implemented with service_category_id as a FK inside of Services.
        - Assume that each service can only belong to a category, but each category can contain one or many services.

<u>ServiceCategories</u>

Records the details of ServiceCategories available in ProService Solutions: artificial intelligence, software development and cybersecurity.

- **Attributes:**
    - service_category_id: INT, AI, UQ, NN, PK
    - name: VARCHAR(100), UQ, NN
    - description: VARCHAR(200), NN
- **Relationship:**
    - A 1:M relationship between ServiceCategories and Services is implemented with service_category_id as a FK inside of Services.
        - Assume that each service can only belong to a category, but each category can contain one or many services.

<u>Requests</u>

Records the details of Requests made by Customers. The main details include date of request and date of completion for better resource management and timely payment processing.

- **Attributes:**
    - request_id: INT, AI, UQ, NN, PK
    - request_date: DATETIME, UQ, NN
    - completion_date: DATETIME
    - customer_id: INT, FK
    - payment_id: INT, FK
- **Relationship:**
    - An optional M:1 relationship between Requests and Customers is implemented with customer_id as a FK inside of Requests.
        - Each request must be associated with a customer, but each customer may have zero or many requests. The customer can be registered in the database but has yet to make a request.
    - An 1:0 relationship between Requests and Payments is implemented with request_id as FK inside of Payments.

        - Assume that each payment only covers one specific request, and each request may have yet to be completed to process payment or paid in one lump sum.
    - An optional N:M relationship between Requests and Services is implemented with service_id and request_id as FKs inside of RequestServices.
        - Each request must consist of one or many services, but each service may be requested zero or many times. There can be a service which has never been requested before, an example being a service that is newly added.
    - A 1:N relationship between Requests and RequestServices (intersection table).

Payments

Records the details of Payments made for the corresponding Requests. The main details include the date of payment and total amount paid. By design, the payment_id will match the request_id in Requests for the corresponding Payments made. As such, payment_id will not have the AI constraint.

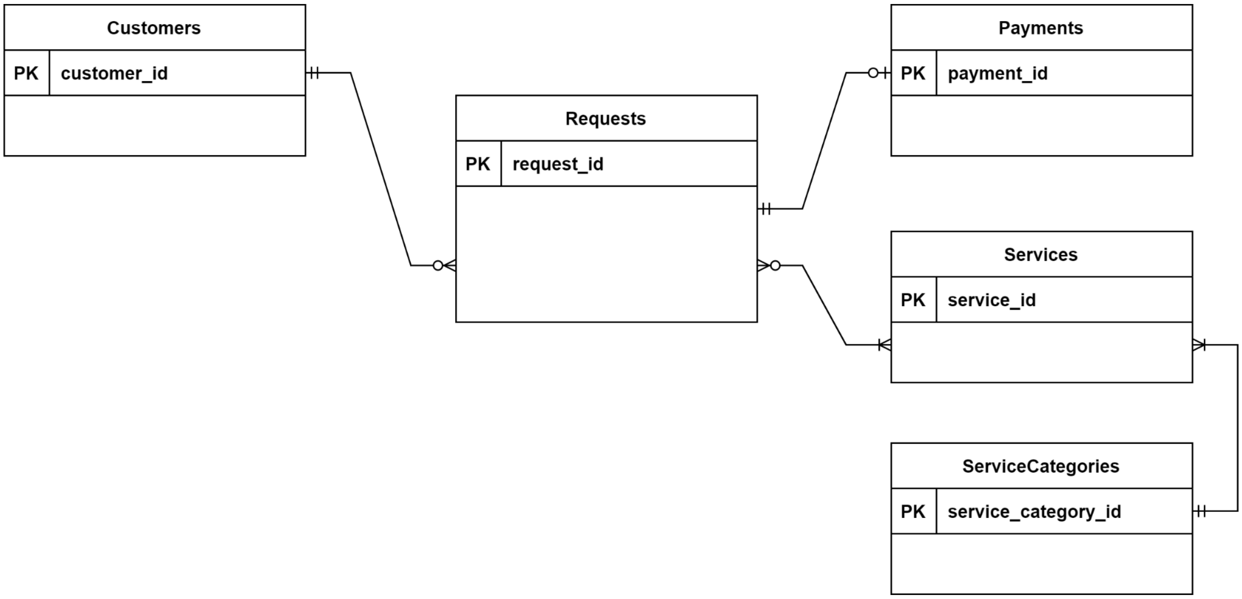
- **Attributes:**
    - payment_id: INT, UQ, NN, PK
    - date: DATETIME, UQ, NN
    - amount: DECIMAL(10, 2), NN
- **Relationship:**
    - An 0:1 relationship between Payments and Requests is implemented with request_id as FK inside of Payments.
        - Assume that each payment only covers one specific request, and each request may have yet to be completed to process payment or paid in one lump sum.

RequestServices

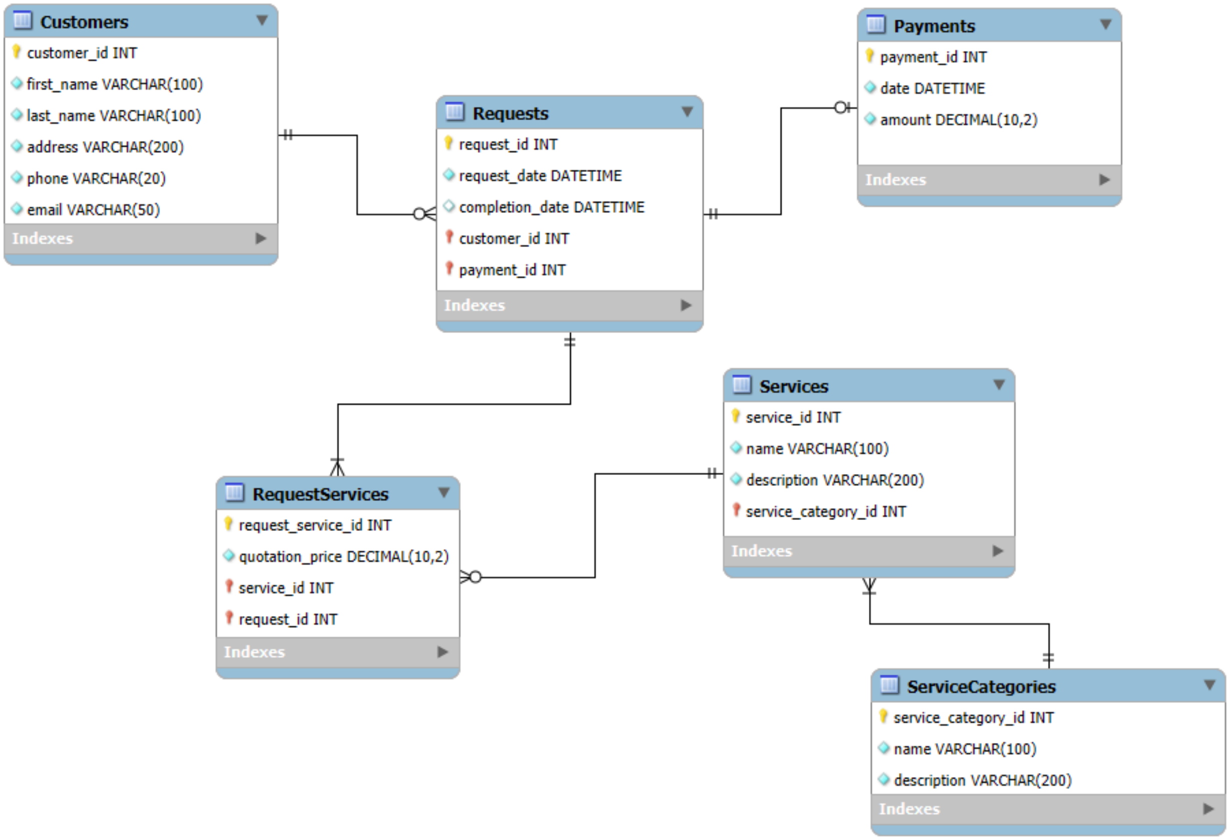
Records the Services involved in the corresponding Requests using an intersection table. The main details include the quotation price. The price of a service offered in each request may be different due to factors such as the size of the customer and the level of customization required.

- **Attributes:**
    - request_service_id: INT, AI, UQ, NN, PK
    - request_id: INT, NN, FK
    - service_id: INT, NN, FK
    - quotation_price: DECIMAL(10, 2), NN
- **Relationship:**
    - A N:1 relationship between RequestServices and Requests.
    - A M:1 relationship between RequestServices and Services.

1.4. Entity-Relationship Diagram



1.5. Schema



1.6. Sample Data

Customers					
customer_id	**first_name**	**last_name**	**address**	**phone**	**email**
1	'James'	'Wheels'	'123 Main St, Atown, USA'	'555-1234'	'james.wheels@gmail.com'
2	'Sam'	'Smith'	'456 Elm St, Btown, USA'	'555-5678'	'sam.smith@gmail.com'
3	'Wendy'	'Button'	'789 Oak St, Ctown, USA'	'555-9012'	'wendy.button@gmail.com'

Services			
service_id	**name**	**description**	**service_category_id**
1	'AI Chatbot Development'	'Develop AI-powered chatbots for customer support'	1
2	'Web Application Development'	'Build custom web applications tailored to client needs'	2
3	'Network Security Audit'	'Conduct audits to assess network security vulnerabilities'	3

ServiceCategories		
service_category_id	**name**	**description**
1	'Artificial Intelligence'	'Advanced AI solutions for businesses'
2	'Software Development'	'Custom software development services'
3	'Cybersecurity'	'Security solutions and services'

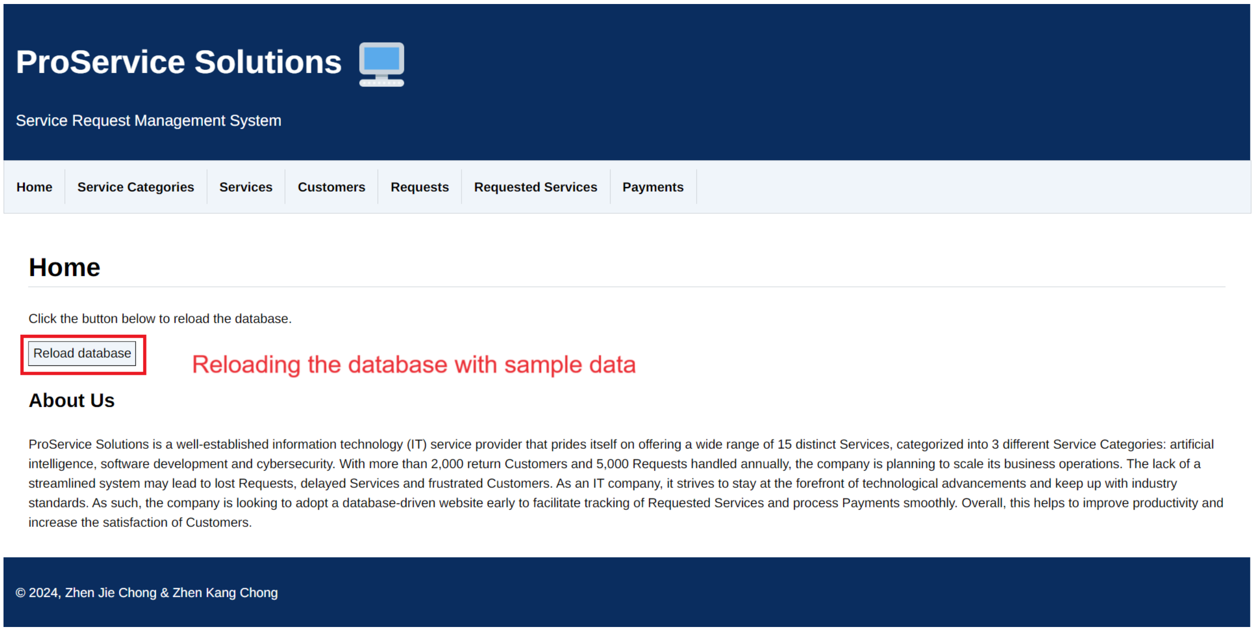
Requests				
request_id	**request_date**	**completion_date**	**customer_id**	**payment_id**
1	'2024-06-15 10:20:00'	'2024-06-17 12:45:30'	1	1
2	'2024-07-15 10:00:00'	'2024-07-18 13:00:00'	1	2
3	'2024-07-16 09:30:00'	'2024-07-23 15:00:00'	2	3
4	'2024-07-17 11:45:00'	NULL	3	NULL

Payments		
payment_id	**date**	**amount**
1	'2024-06-19 18:23:56'	780.25
2	'2024-07-19 11:30:00'	1000.75
3	'2024-07-24 14:00:00'	2300.50

RequestServices			
request_service_id	**request_id**	**service_id**	**quotation_price**
1	1	2	780.25
2	2	1	1000.75
3	3	2	1500.00
4	3	3	800.50
5	4	3	1200.00

1.7. UI Screenshots

Home page (reloading the database with sample data):



ProService Solutions

Service Request Management System

Home

Click the button below to reload the database.

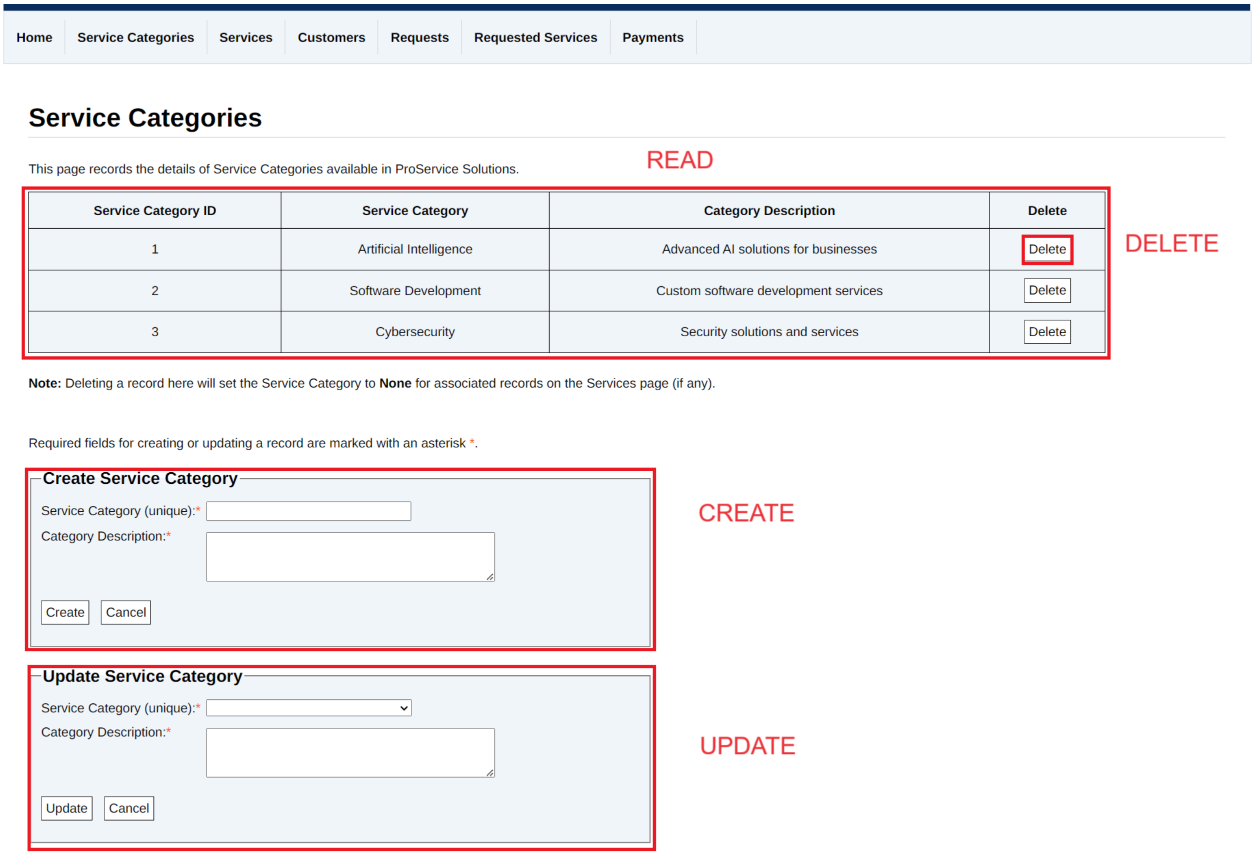
Reloading the database with sample data

About Us

ProService Solutions is a well-established information technology (IT) service provider that prides itself on offering a wide range of 15 distinct Services, categorized into 3 different Service Categories: artificial intelligence, software development and cybersecurity. With more than 2,000 return Customers and 5,000 Requests handled annually, the company is planning to scale its business operations. The lack of a streamlined system may lead to lost Requests, delayed Services and frustrated Customers. As an IT company, it strives to stay at the forefront of technological advancements and keep up with industry standards. As such, the company is looking to adopt a database-driven website early to facilitate tracking of Requested Services and process Payments smoothly. Overall, this helps to improve productivity and increase the satisfaction of Customers.

© 2024, Zhen Jie Chong & Zhen Kang Chong

Service Categories page (CREATE/READ/UPDATE/DELETE):



Service Categories

This page records the details of Service Categories available in ProService Solutions.

Service Category ID	Service Category	Category Description	Delete
1	Artificial Intelligence	Advanced AI solutions for businesses	Delete
2	Software Development	Custom software development services	Delete
3	Cybersecurity	Security solutions and services	Delete

Note: Deleting a record here will set the Service Category to **None** for associated records on the Services page (if any).

Required fields for creating or updating a record are marked with an asterisk *.

Create Service Category

Service Category (unique):*

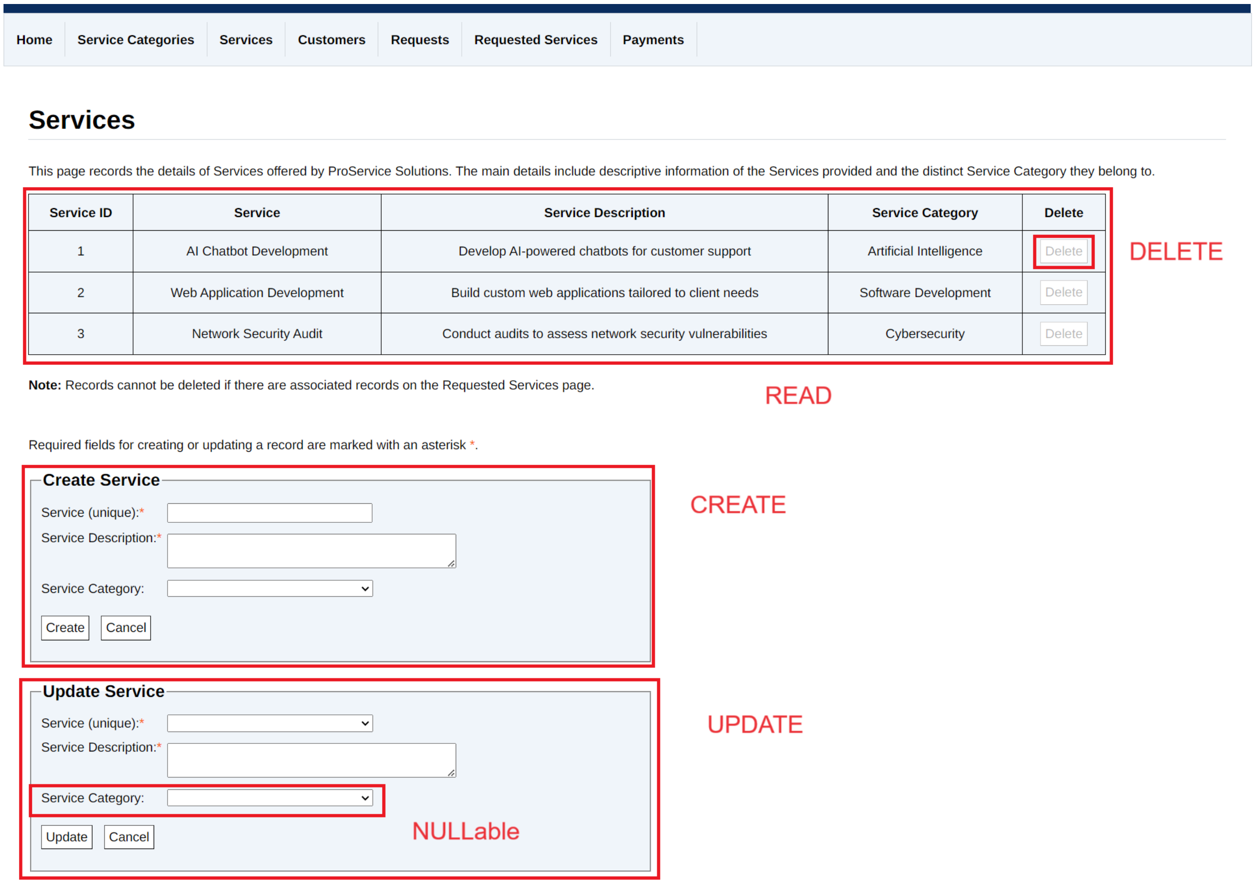
Category Description:*

Update Service Category

Service Category (unique):*

Category Description:*

Services page (CREATE/READ/UPDATE/DELETE):



Services

This page records the details of Services offered by ProService Solutions. The main details include descriptive information of the Services provided and the distinct Service Category they belong to.

Service ID	Service	Service Description	Service Category	Delete
1	AI Chatbot Development	Develop AI-powered chatbots for customer support	Artificial Intelligence	Delete
2	Web Application Development	Build custom web applications tailored to client needs	Software Development	Delete
3	Network Security Audit	Conduct audits to assess network security vulnerabilities	Cybersecurity	Delete

DELETE

Note: Records cannot be deleted if there are associated records on the Requested Services page.

READ

Required fields for creating or updating a record are marked with an asterisk *.

Create Service

Service (unique):*

Service Description:*

Service Category:

CREATE

Update Service

Service (unique):*

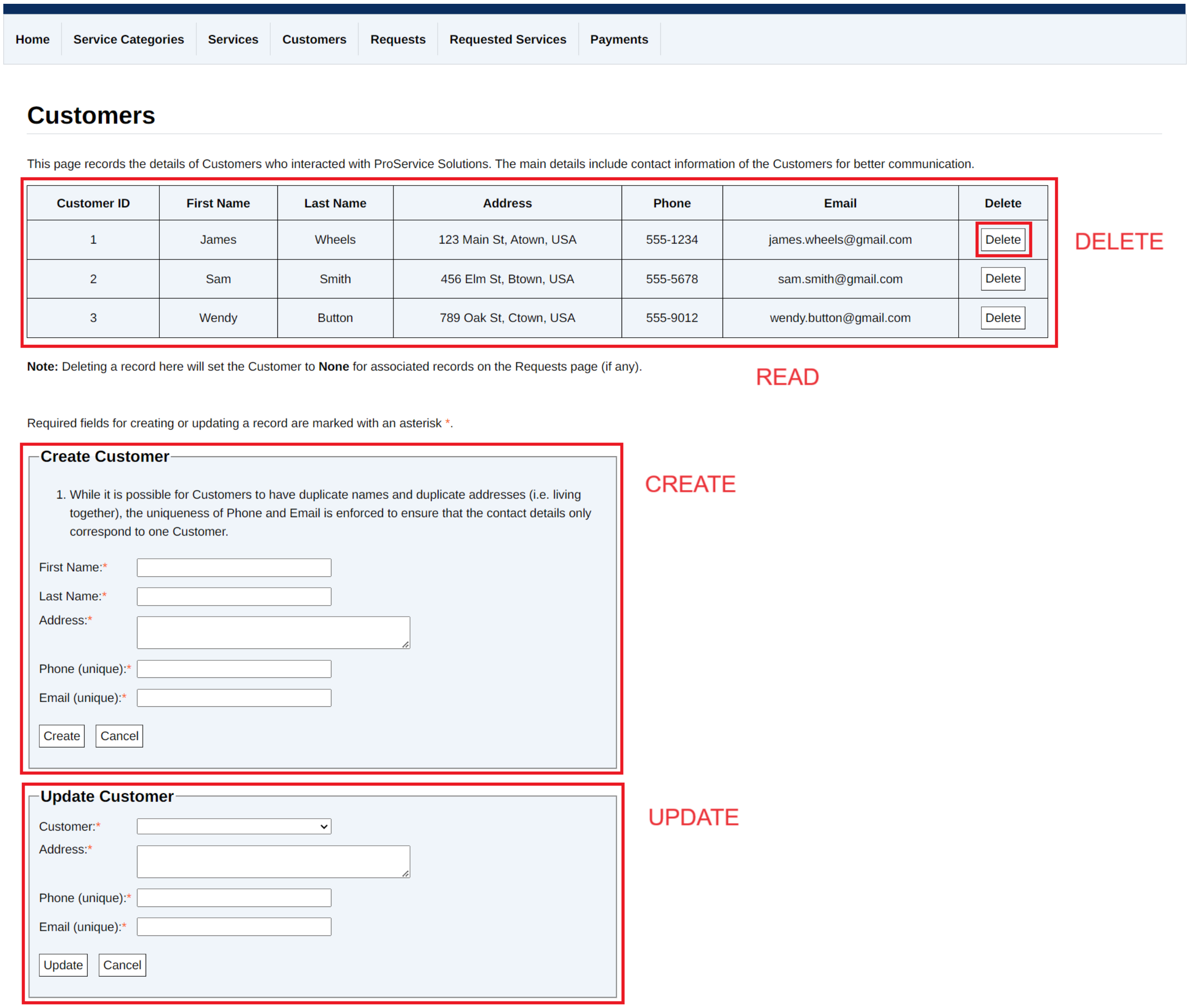
Service Description:*

Service Category:

NULLable

UPDATE

Customers page (CREATE/READ/UPDATE/DELETE):



Customers

This page records the details of Customers who interacted with ProService Solutions. The main details include contact information of the Customers for better communication.

Customer ID	First Name	Last Name	Address	Phone	Email	Delete
1	James	Wheels	123 Main St, Atown, USA	555-1234	james.wheels@gmail.com	Delete
2	Sam	Smith	456 Elm St, Btown, USA	555-5678	sam.smith@gmail.com	Delete
3	Wendy	Button	789 Oak St, Ctown, USA	555-9012	wendy.button@gmail.com	Delete

Note: Deleting a record here will set the Customer to **None** for associated records on the Requests page (if any).

Required fields for creating or updating a record are marked with an asterisk *.

Create Customer

1. While it is possible for Customers to have duplicate names and duplicate addresses (i.e. living together), the uniqueness of Phone and Email is enforced to ensure that the contact details only correspond to one Customer.

First Name:*

Last Name:*

Address:*

Phone (unique):*

Email (unique):*

Update Customer

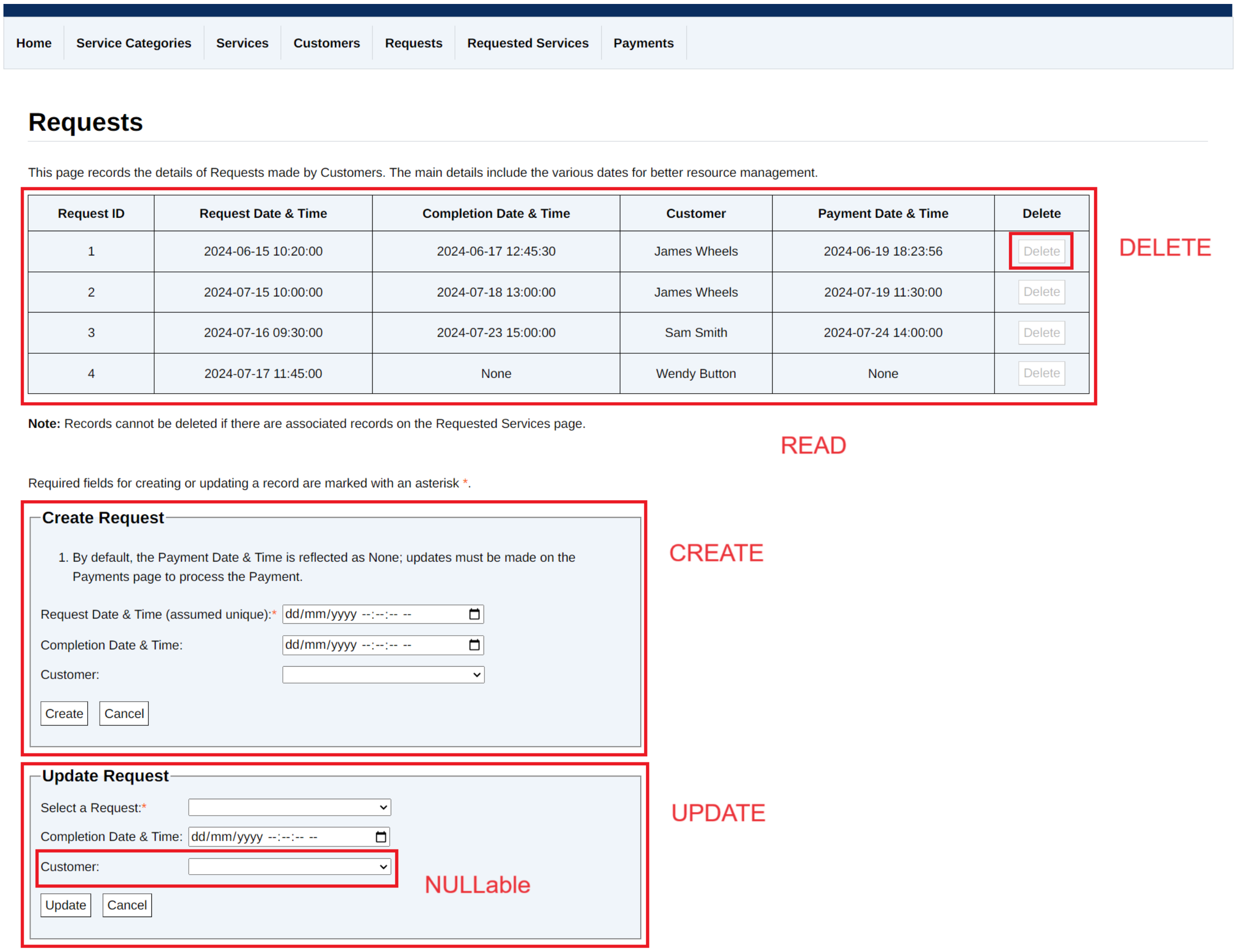
Customer:*

Address:*

Phone (unique):*

Email (unique):*

Requests page (CREATE/READ/UPDATE/DELETE):



Requests

This page records the details of Requests made by Customers. The main details include the various dates for better resource management.

Request ID	Request Date & Time	Completion Date & Time	Customer	Payment Date & Time	Delete
1	2024-06-15 10:20:00	2024-06-17 12:45:30	James Wheels	2024-06-19 18:23:56	Delete
2	2024-07-15 10:00:00	2024-07-18 13:00:00	James Wheels	2024-07-19 11:30:00	Delete
3	2024-07-16 09:30:00	2024-07-23 15:00:00	Sam Smith	2024-07-24 14:00:00	Delete
4	2024-07-17 11:45:00	None	Wendy Button	None	Delete

DELETE

Note: Records cannot be deleted if there are associated records on the Requested Services page.

READ

Required fields for creating or updating a record are marked with an asterisk *.

Create Request

1. By default, the Payment Date & Time is reflected as None; updates must be made on the Payments page to process the Payment.

Request Date & Time (assumed unique):* dd/mm/yyyy --:--:-- --

Completion Date & Time: dd/mm/yyyy --:--:-- --

Customer:

CREATE

Update Request

Select a Request:*

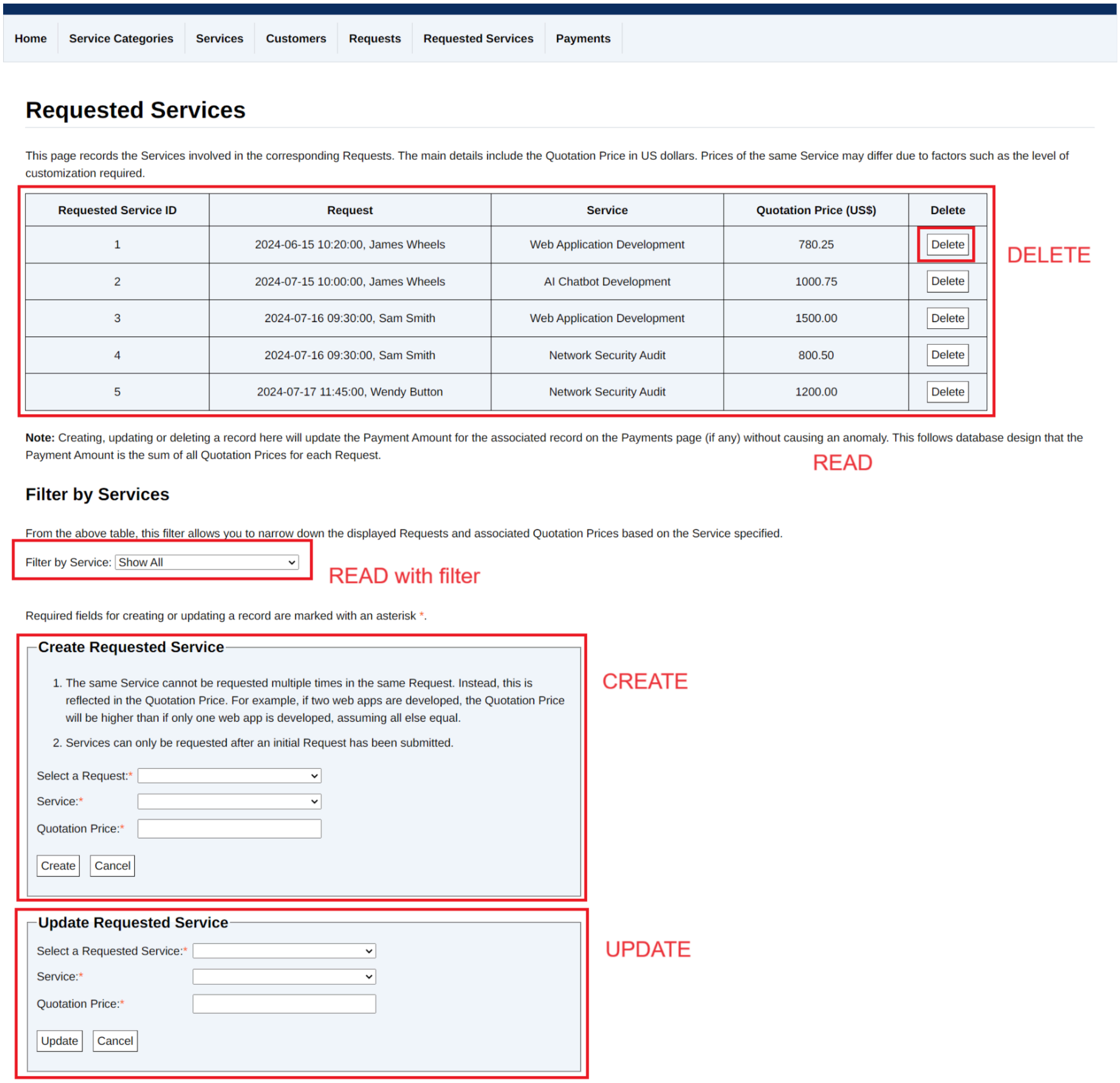
Completion Date & Time: dd/mm/yyyy --:--:-- --

Customer:

NULLable

UPDATE

Requested Services page (CREATE/READ/UPDATE/DELETE):



Requested Services

This page records the Services involved in the corresponding Requests. The main details include the Quotation Price in US dollars. Prices of the same Service may differ due to factors such as the level of customization required.

Requested Service ID	Request	Service	Quotation Price (US$)	Delete
1	2024-06-15 10:20:00, James Wheels	Web Application Development	780.25	Delete
2	2024-07-15 10:00:00, James Wheels	AI Chatbot Development	1000.75	Delete
3	2024-07-16 09:30:00, Sam Smith	Web Application Development	1500.00	Delete
4	2024-07-16 09:30:00, Sam Smith	Network Security Audit	800.50	Delete
5	2024-07-17 11:45:00, Wendy Button	Network Security Audit	1200.00	Delete

DELETE

Note: Creating, updating or deleting a record here will update the Payment Amount for the associated record on the Payments page (if any) without causing an anomaly. This follows database design that the Payment Amount is the sum of all Quotation Prices for each Request.

READ

Filter by Services

From the above table, this filter allows you to narrow down the displayed Requests and associated Quotation Prices based on the Service specified.

Filter by Service: Show All

READ with filter

Required fields for creating or updating a record are marked with an asterisk *.

Create Requested Service

1. The same Service cannot be requested multiple times in the same Request. Instead, this is reflected in the Quotation Price. For example, if two web apps are developed, the Quotation Price will be higher than if only one web app is developed, assuming all else equal.
2. Services can only be requested after an initial Request has been submitted.

Select a Request:*

Service:*

Quotation Price:*

Create Cancel

CREATE

Update Requested Service

Select a Requested Service:*

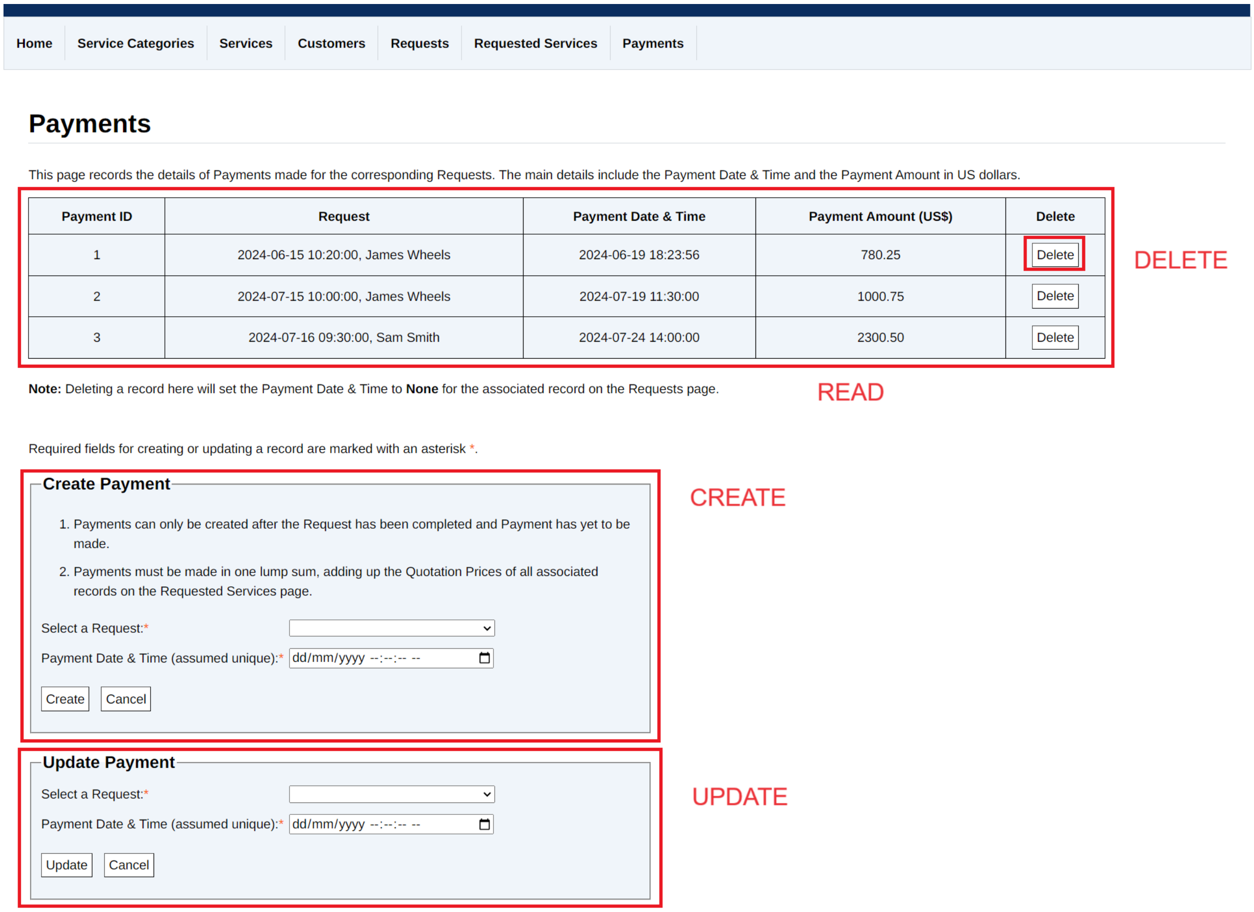
Service:*

Quotation Price:*

Update Cancel

UPDATE

Payments page (CREATE/READ/UPDATE/DELETE):



Payments

This page records the details of Payments made for the corresponding Requests. The main details include the Payment Date & Time and the Payment Amount in US dollars.

Payment ID	Request	Payment Date & Time	Payment Amount (US$)	Delete
1	2024-06-15 10:20:00, James Wheels	2024-06-19 18:23:56	780.25	Delete
2	2024-07-15 10:00:00, James Wheels	2024-07-19 11:30:00	1000.75	Delete
3	2024-07-16 09:30:00, Sam Smith	2024-07-24 14:00:00	2300.50	Delete

DELETE

Note: Deleting a record here will set the Payment Date & Time to **None** for the associated record on the Requests page.

READ

Required fields for creating or updating a record are marked with an asterisk *.

Create Payment

1. Payments can only be created after the Request has been completed and Payment has yet to be made.
2. Payments must be made in one lump sum, adding up the Quotation Prices of all associated records on the Requested Services page.

Select a Request:*

Payment Date & Time (assumed unique):* dd/mm/yyyy --:--:-- --

CREATE

Update Payment

Select a Request:*

Payment Date & Time (assumed unique):* dd/mm/yyyy --:--:-- --

UPDATE