

End to End with Testcontainers and Playwright



Hi! 🖐️

My name is Vemund

Raise your hand if you have...

Raise your hand if you have...

... have good unit-test coverage in your
codebase

Raise your hand if you have...

... have good unit-test coverage in your codebase

... have good end-to-end coverage in your codebase



Raise your hand if you...

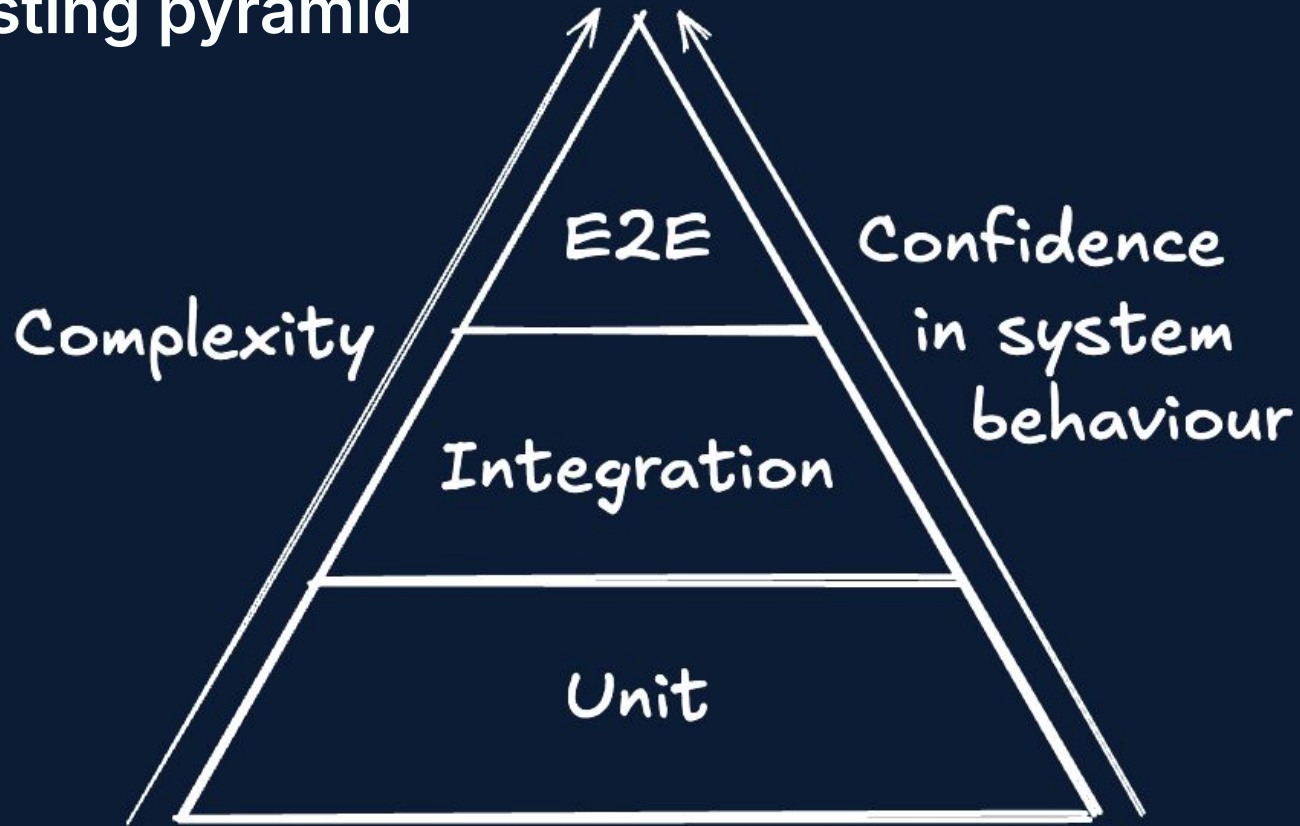
... have good unit-test coverage in your codebase

... have good end-to-end coverage in your codebase

... already use Testcontainers and Playwright

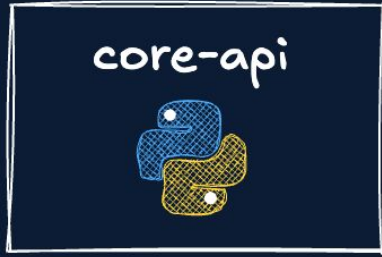


The testing pyramid

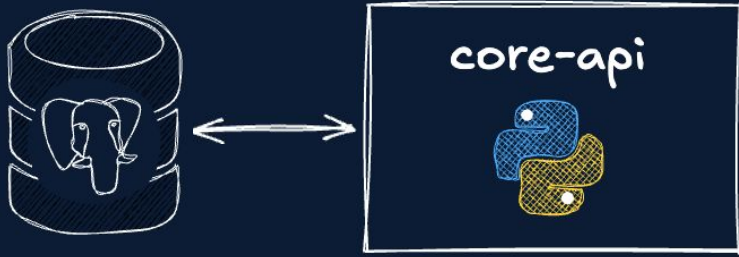


Our test environment

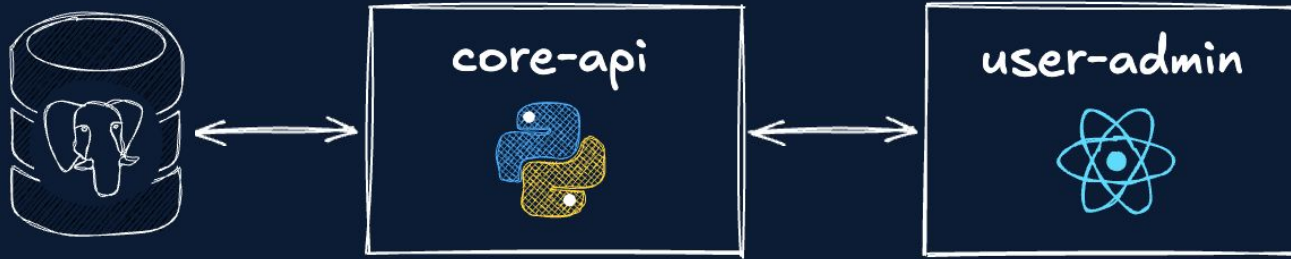
Our test environment



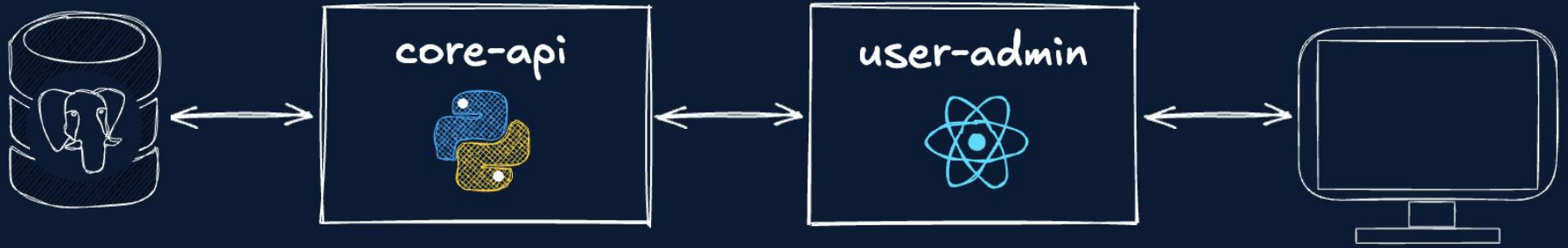
Our test environment



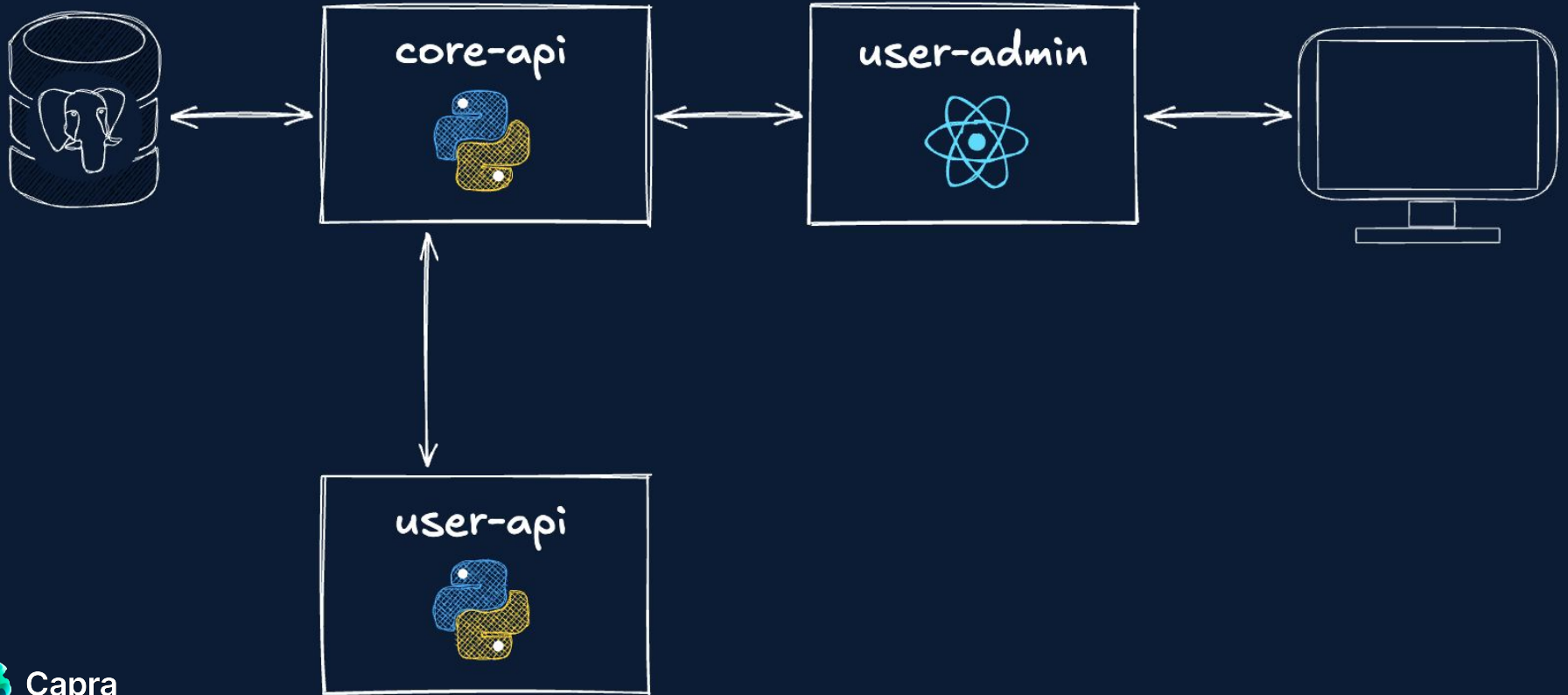
Our test environment



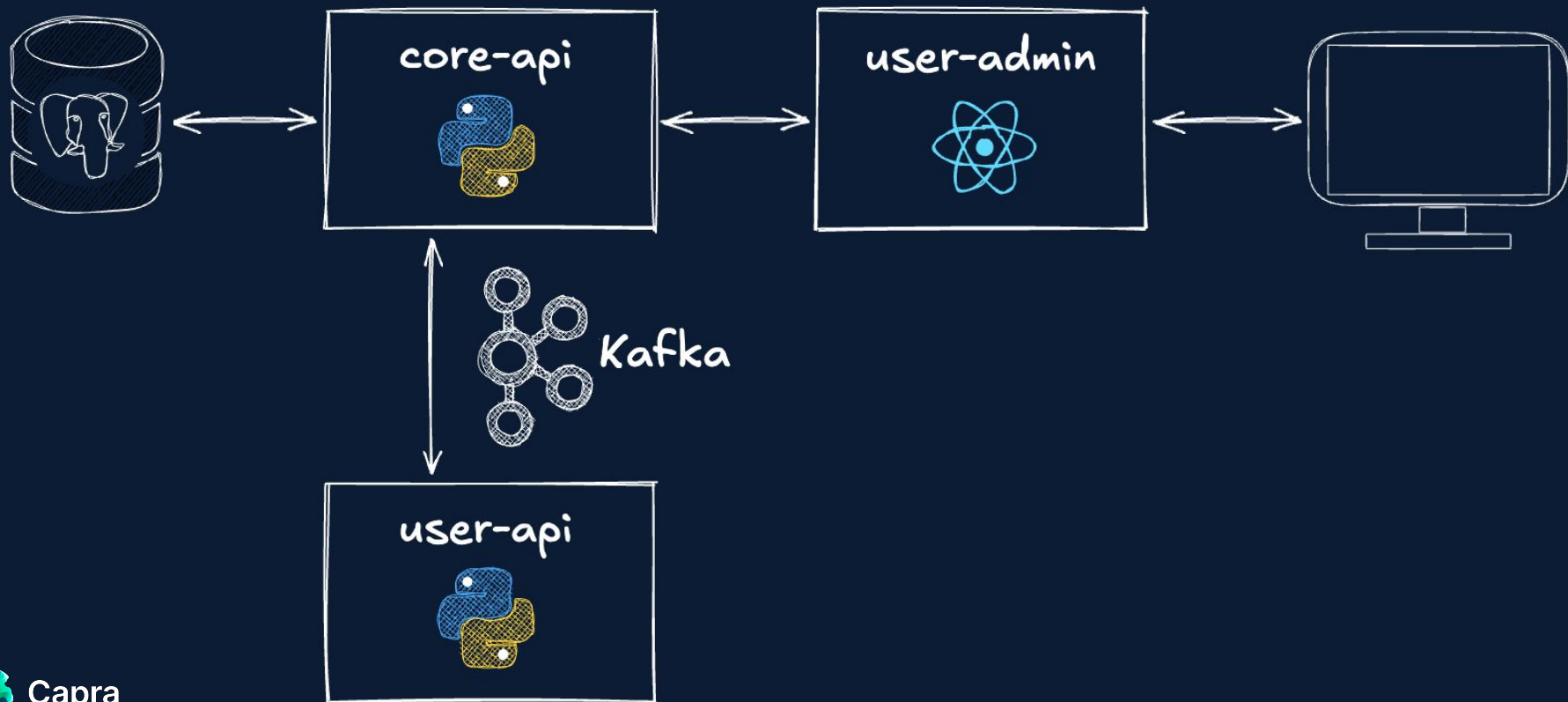
Our test environment



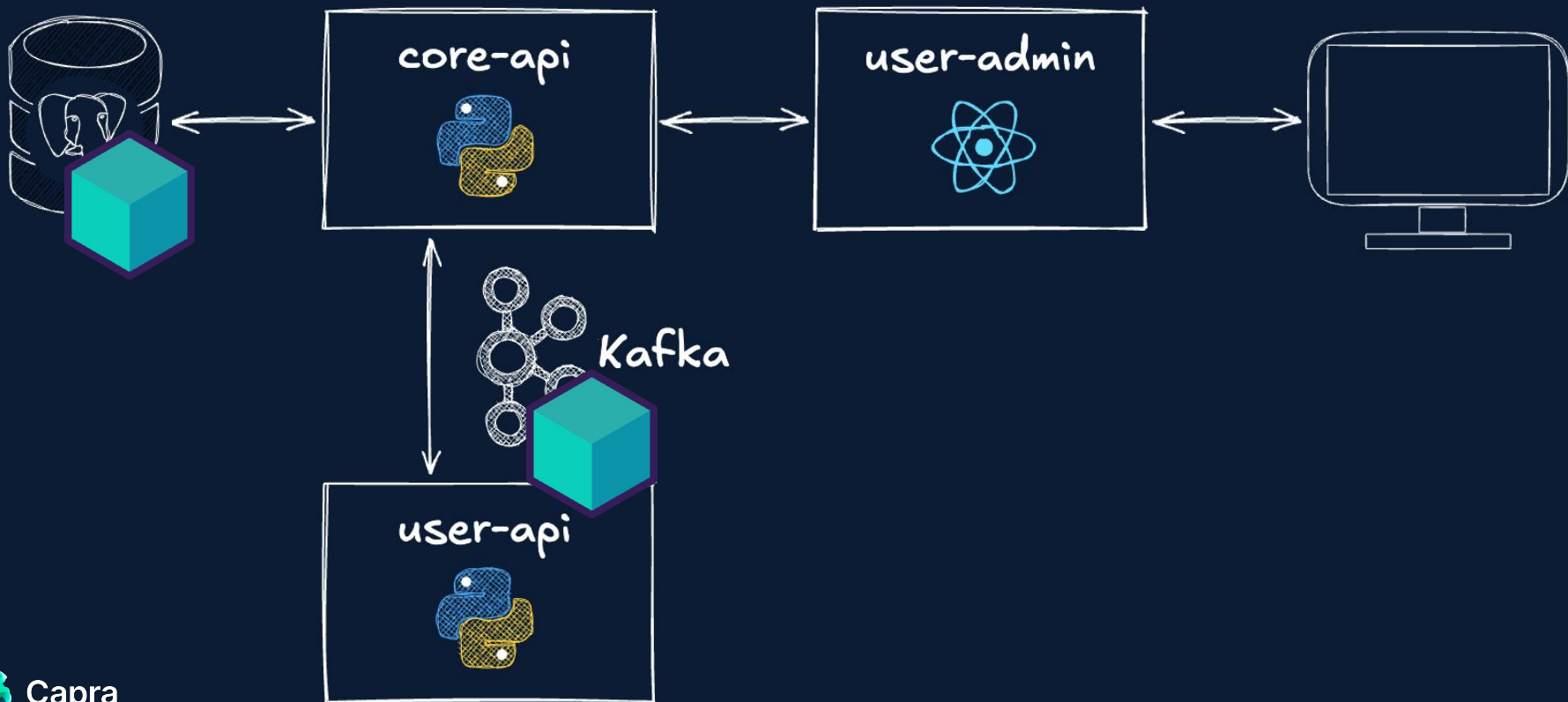
Our test environment



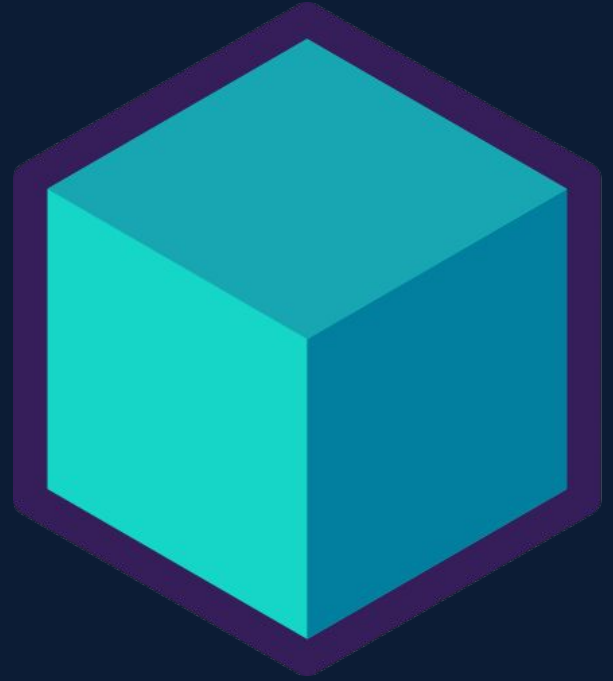
Our test environment



Our test environment

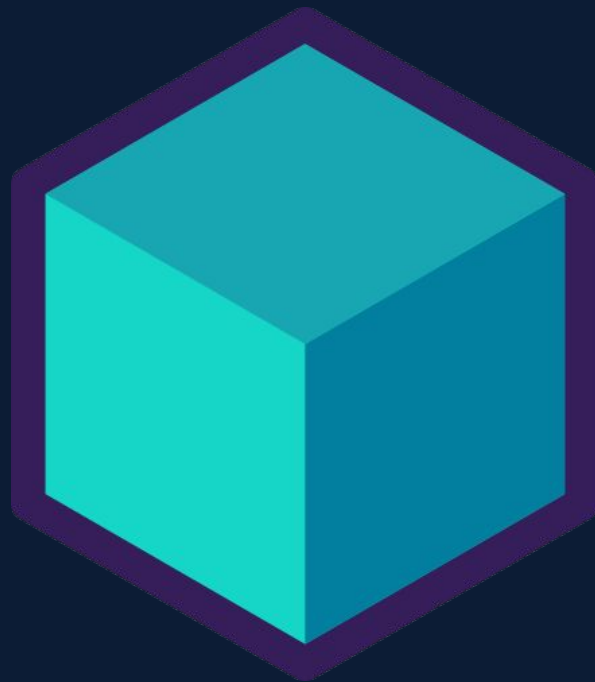


Why Testcontainers?



Why Testcontainers?

- Reduces risk by running actual implementations of external services



Why Testcontainers?

- Reduces risk by running actual implementations of external

```
CREATE TABLE "user" (  
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),  
  name TEXT NOT NULL  
)
```

```
WITH case_ids AS (  
  SELECT DISTINCT case_dialogue.case_id  
  FROM case_dialogue_message  
  LEFT JOIN case_dialogue ON (case_dialogue_message.case_dialogue_id = case_dialogue.id)  
  WHERE case_dialogue_message.agent_id = :user_id
```

UNION

```
  SELECT DISTINCT case_comment.case_id  
  FROM case_comment  
  WHERE case_comment.created_by_id = :user_id
```

UNION

```
  SELECT DISTINCT case_note.case_id  
  FROM case_note  
  WHERE case_note.created_by_id = :user_id
```

UNION

```
  SELECT DISTINCT "case".id  
  FROM "case"  
  WHERE "case".responsible_id = :user_id
```

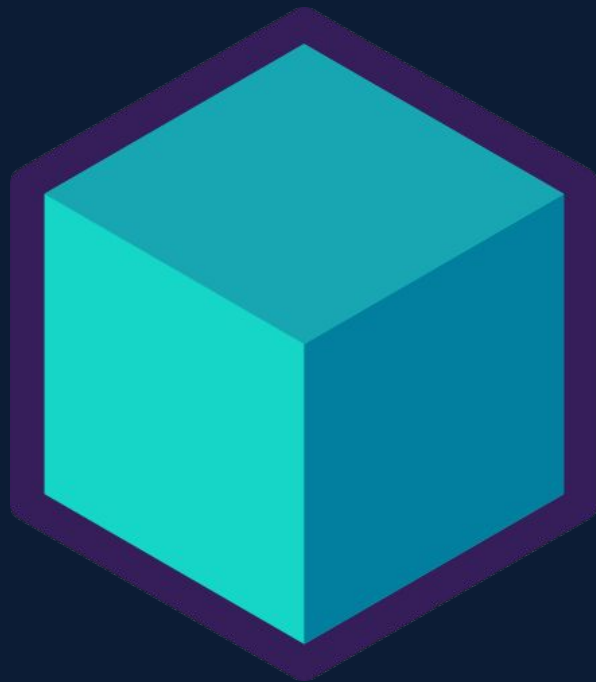
```

JOIN LATERAL (
    SELECT coalesce(jsonb_agg(jsonb_set(to_jsonb(case_dialogue), '{messages}', messages), '{events}', events)), '[]') AS
FROM case_dialogue
LEFT JOIN LATERAL (
    SELECT coalesce(jsonb_agg(jsonb_set(to_jsonb(case_dialogue_message), '{attachments}', attachments), '{agent}', co
FROM case_dialogue_message
LEFT JOIN "user" agent ON agent.id = case_dialogue_message.agent_id AND agent.pta = :pta
LEFT JOIN LATERAL (
    SELECT coalesce(jsonb_agg(attachment), '[]') AS attachments
FROM attachment
    INNER JOIN attachment_to_case_dialogue_message
    ON attachment_to_case_dialogue_message.attachment_id = attachment.id
    WHERE case_dialogue_message.id = attachment_to_case_dialogue_message.case_dialogue_message_id) AS attachments ON TRUE
LEFT JOIN LATERAL (
    SELECT coalesce(jsonb_agg(case_dialogue_event), '[]') AS events
FROM case_dialogue_event
    WHERE case_dialogue.id = case_dialogue_event.case_dialogue_id) AS events ON TRUE
LEFT JOIN LATERAL (
    SELECT coalesce(jsonb_agg(jsonb_set(to_jsonb(case_note), '{created_by}', to_jsonb(created_by_user))), '[]') AS notes
FROM case_note
LEFT JOIN "user" created_by_user ON created_by_user.id = case_note.created_by_id AND created
    WHERE "case".id = case_note.case_id) AS notes ON TRUE
    WHERE "case".pta = :pta AND "case".access_group = :access_group
FROM "case"

```

Why Testcontainers?

- Reduces risk by running actual implementations of external services
- Single-line provisioning



Why Testcontainers?

- Reduces risk by running actual implementations of external services
- Single-line provisioning

```
postgres = PostgresContainer("postgres:17").start()
```



Why Testcontainers?

- Reduces risk by running actual implementations of external services
- Single-line provisioning
- Enables fully local development



Why Testcontainers?

- Reduces risk by running actual implementations of external services
- Single-line provisioning
- Enables fully local development
- Integrates with test lifecycles



Why Testcontainers?

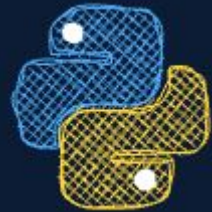
- Reduces risk by running actual implementations of external services

```
- @pytest.fixture(scope="session")
  def postgres():
- E     postgres = PostgresContainer("postgres:17").start()
- I     yield postgres
      postgres.stop()
```



How does it work?

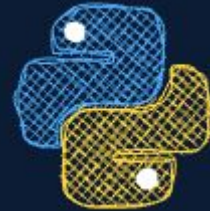
How does it work?



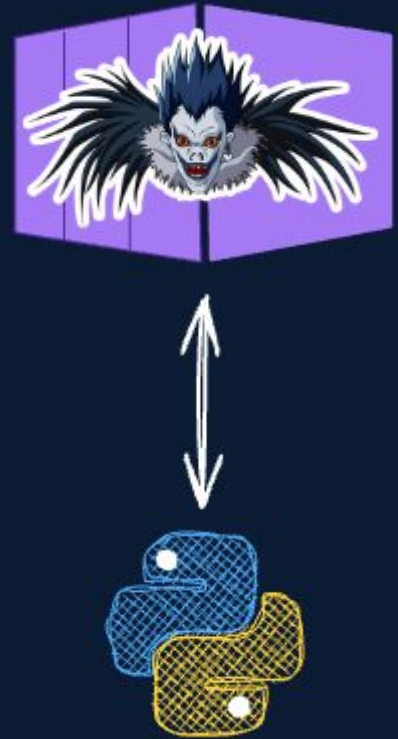
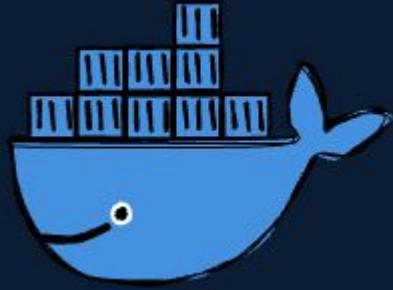
How does it work?



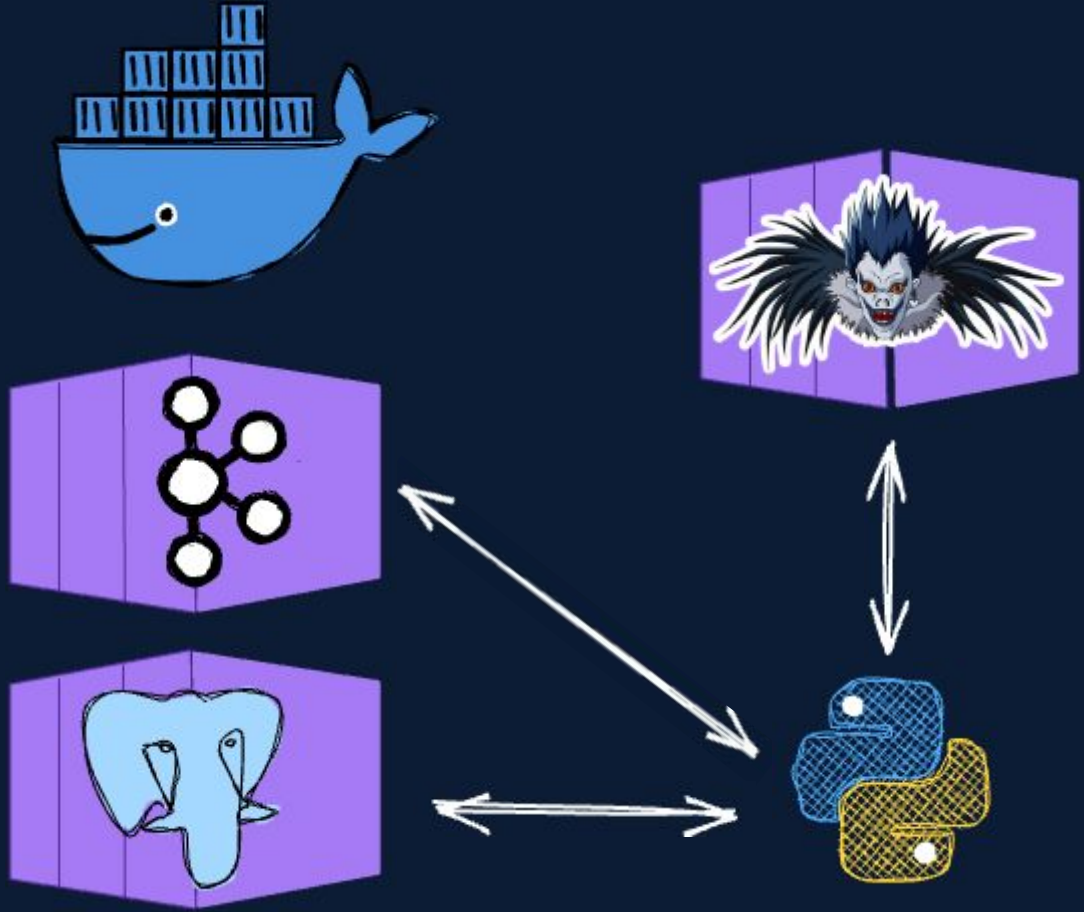
How does it work?



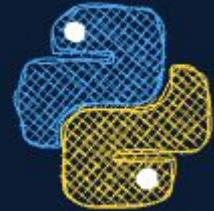
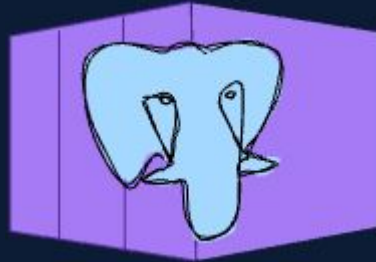
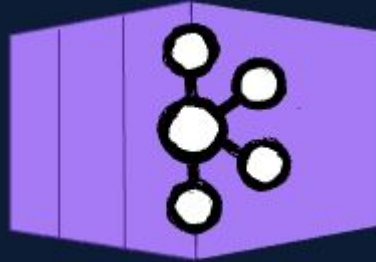
How does it work?



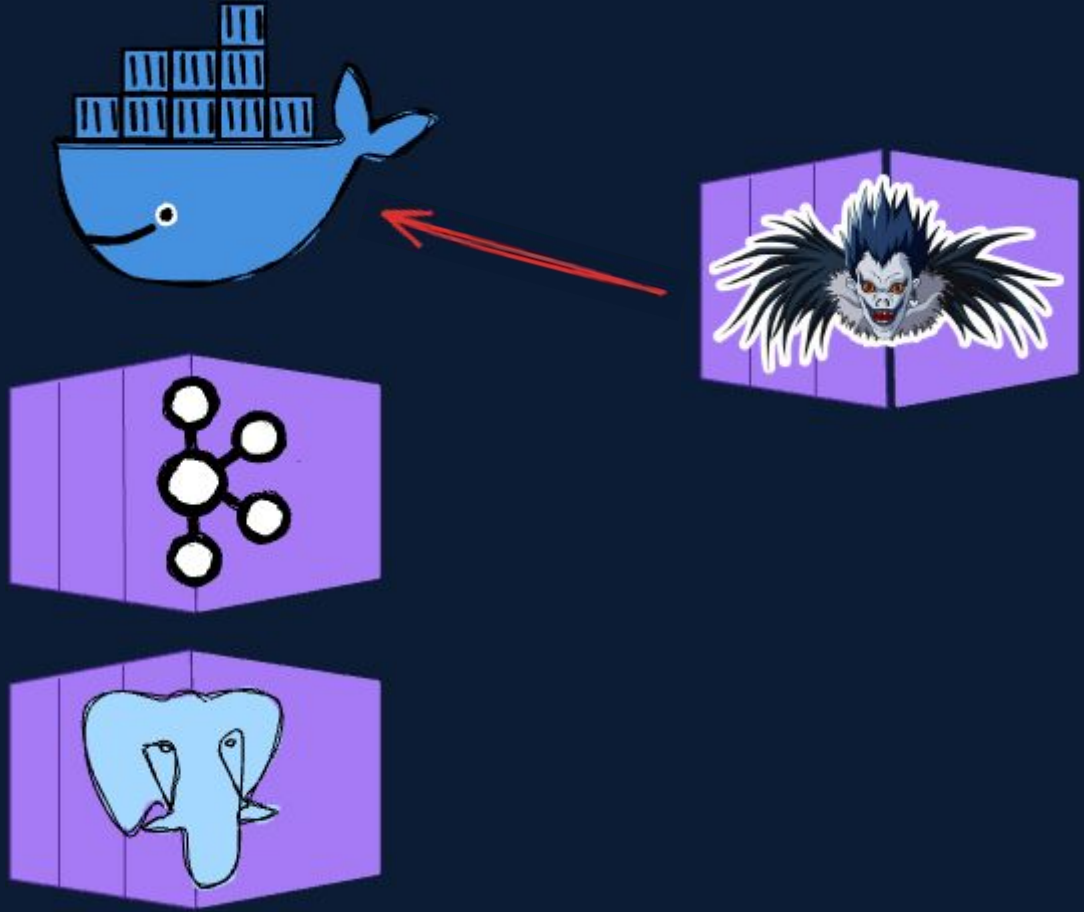
How does it work?



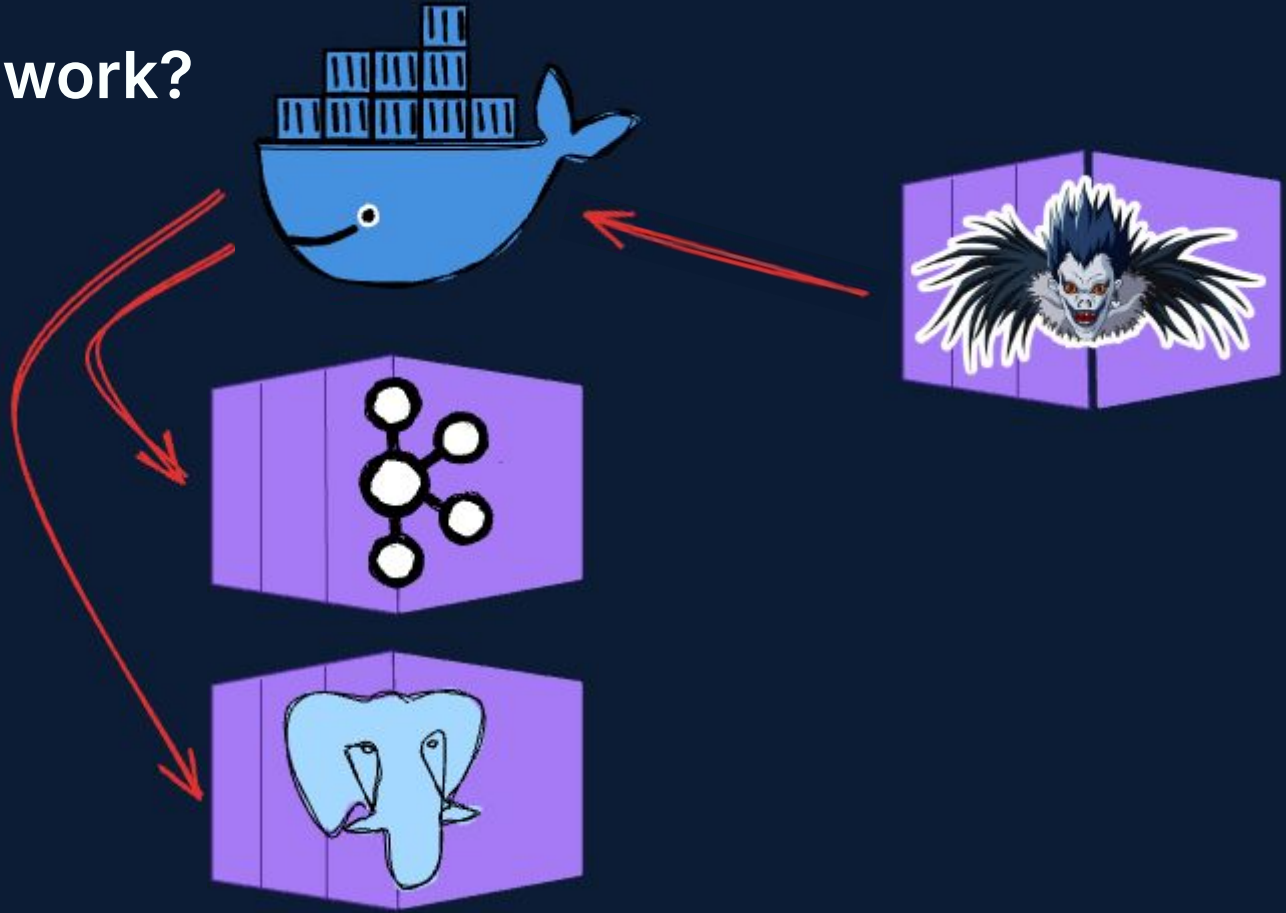
How does it work?



How does it work?



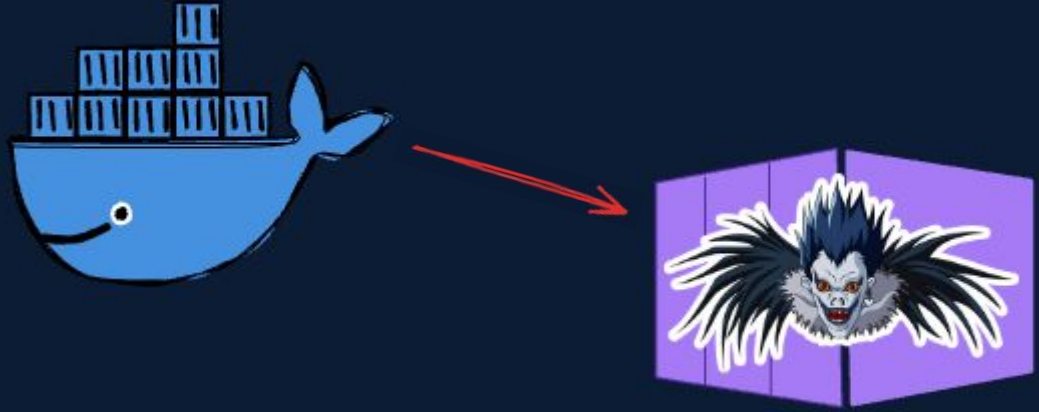
How does it work?



How does it work?



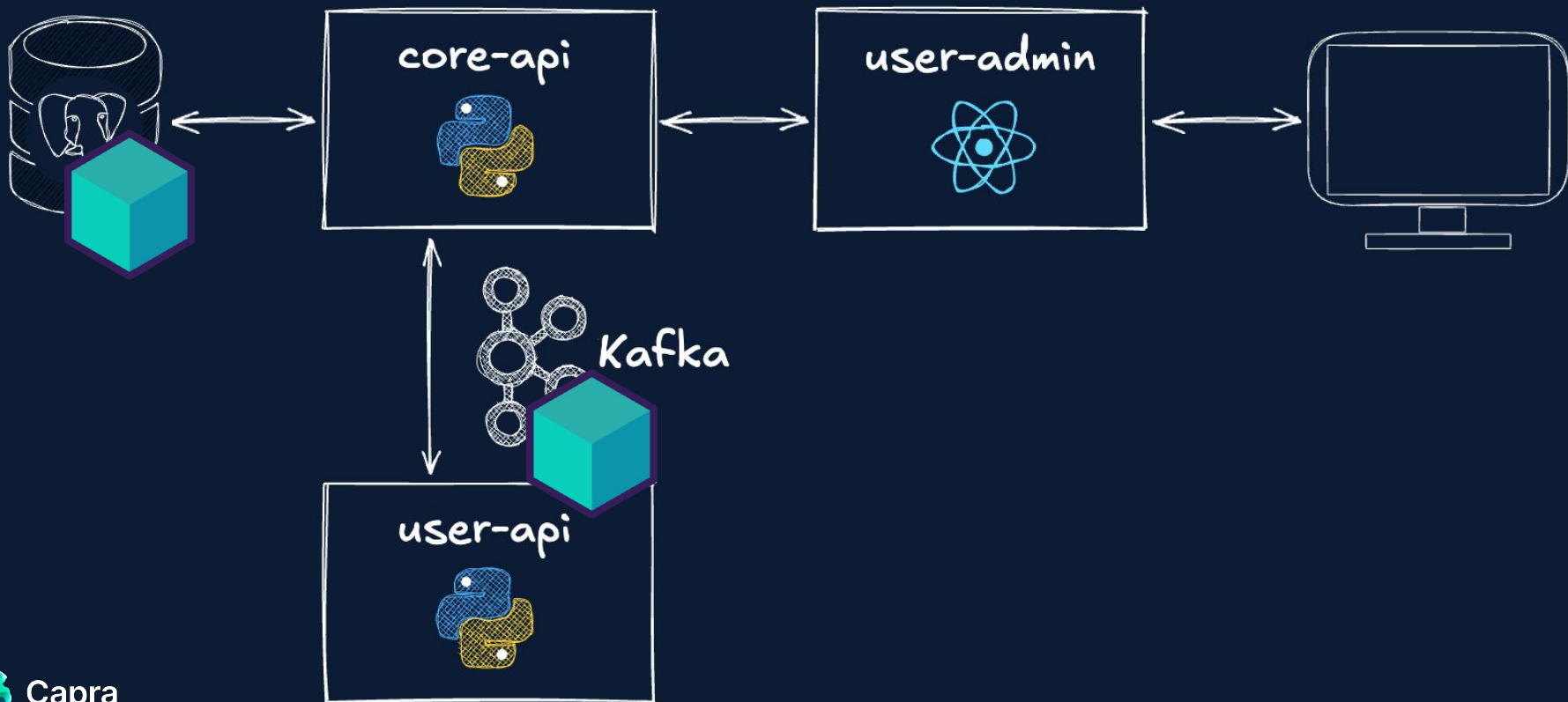
How does it work?



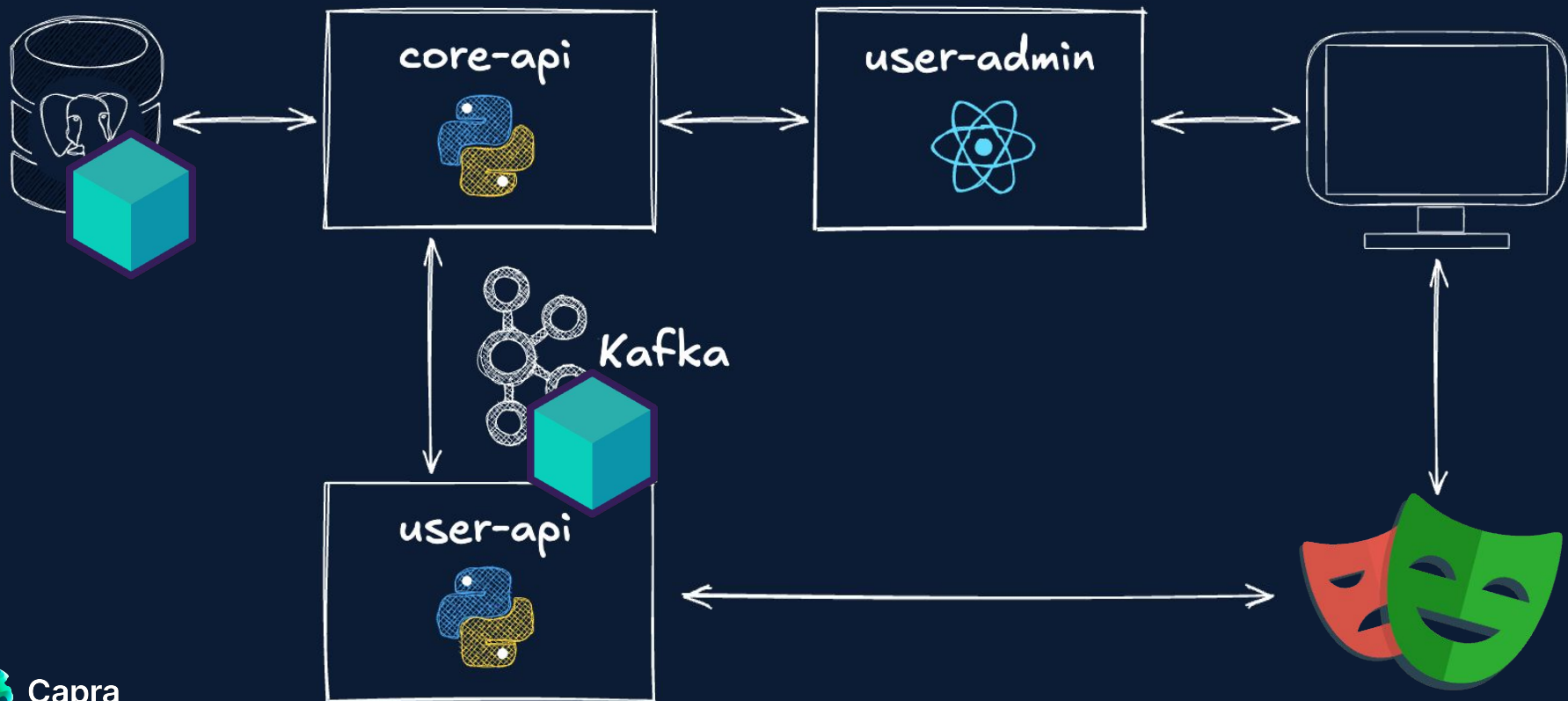
How does it work?



Our test environment



Our test environment



What is Playwright?



What is Playwright?

- Browser automation library



What is Playwright?

- Browser automation library
- Available in multiple languages



What is Playwright?

- Browser automation library
- Available in multiple languages
- Supports Chromium, Firefox, Webkit, Edge



What is Playwright?

- Browser automation library
- Available in multiple languages
- Supports Chromium, Firefox, Webkit, Edge
- Can generate tests by recording user input

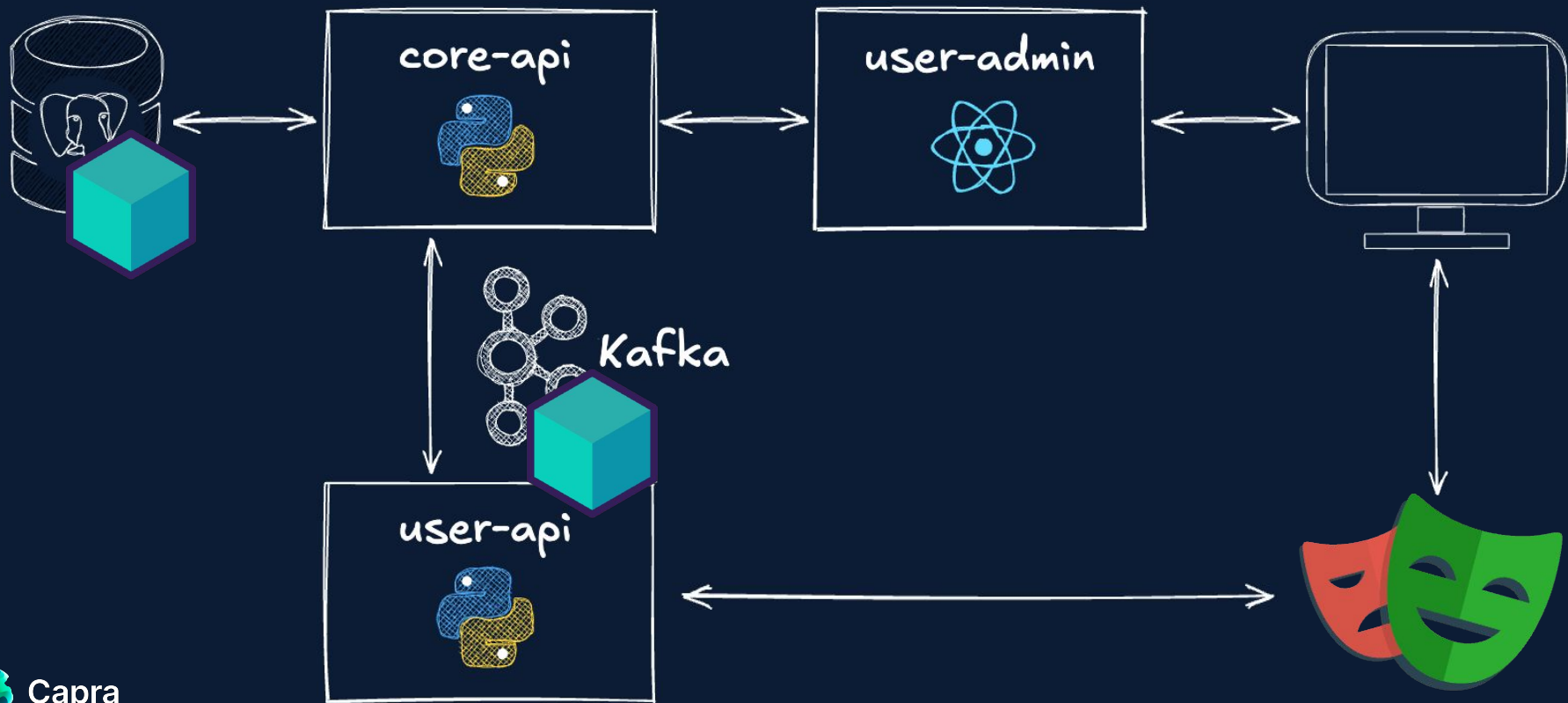


What is Playwright?

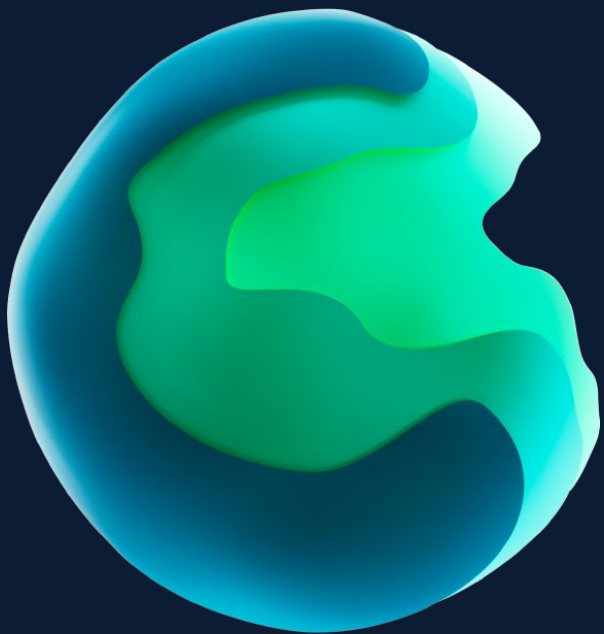
- Browser automation library
- Available in multiple languages
- Supports Chromium, Firefox, Webkit, Edge
- Can generate tests by recording user input
- Integrates with pytest



Our test environment



Live demo



Thank you!

 vemund@santi.no

 <https://github.com/santi/pycon-ireland-2025>

 <https://www.linkedin.com/in/vemundsanti>

