

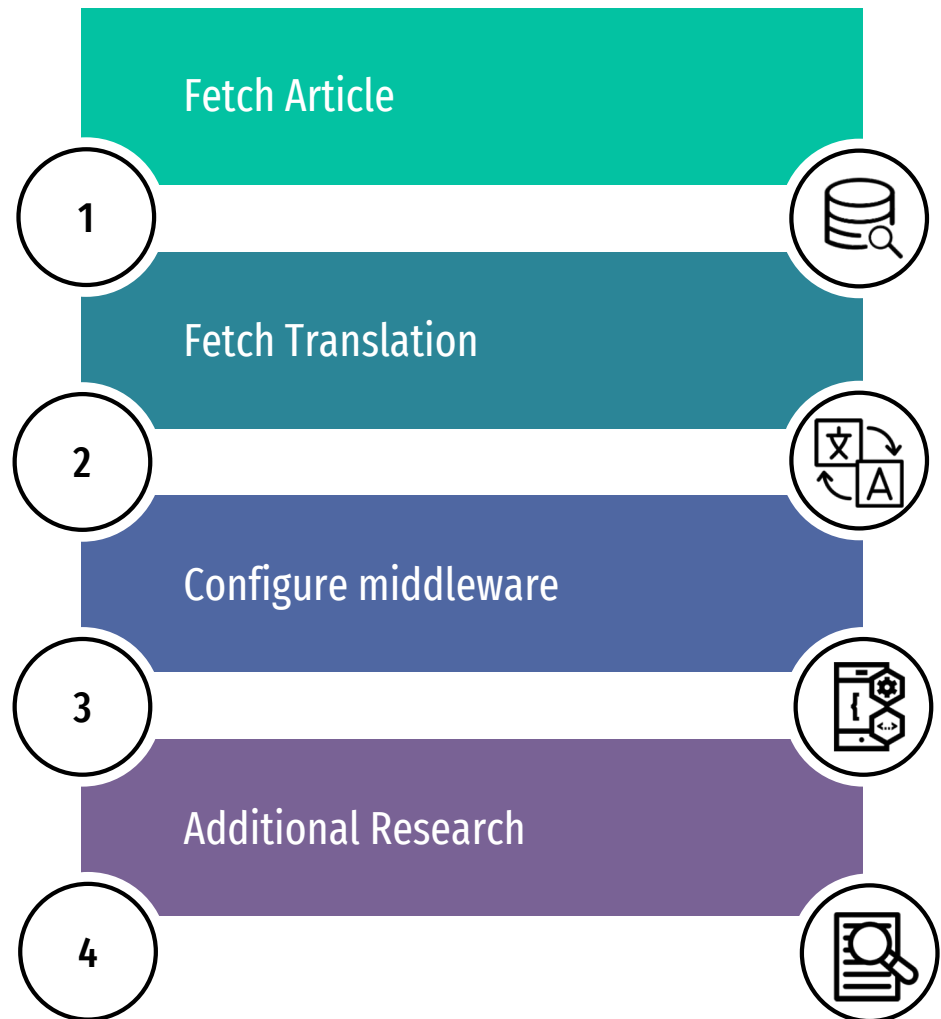


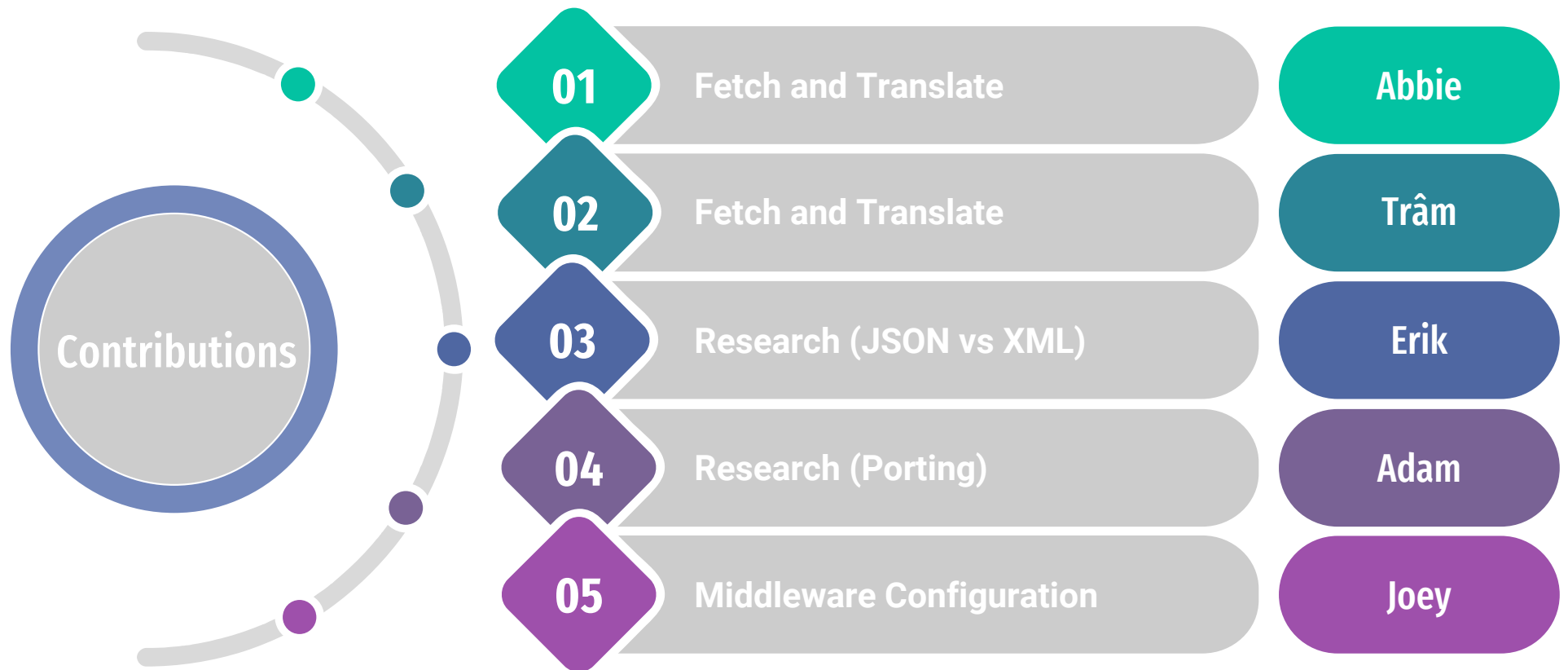
# Project Symmetry

🧡 Team Scrumptious 🧡

API & Middleware

# Accomplishments





## User Story 1



As a front-end developer, I want to be able to retrieve articles from Wikipedia via API, so that I can obtain data for translation comparisons.

Size: 3

```
@app.get("/get_article", response_model=SourceArticleResponse)
def get_article(url: str = Query(None), title: str = Query(None)):
    logging.info("Calling get article endpoint")

    if url:
        title = extract_title_from_url(url)
        if not title:
            logging.info("Invalid Wikipedia URL provided.")
            raise HTTPException(status_code=400, detail="Invalid Wikipedia URL provided.")

    if not title:
        logging.info("Either 'url' or 'title' must be provided.")
        raise HTTPException(status_code=400, detail="Either 'url' or 'title' must be provided.")

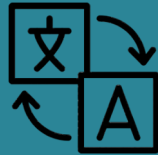
    page = wiki_wiki.page(title)

    if not page.exists():
        logging.info("Article not found.")
        raise HTTPException(status_code=404, detail="Article not found.")

    article_content = page.text  # Get the article text

    # Fetch available languages
    languages = list(page.langlinks.keys())

    return {"source_article": article_content, "article_languages": languages}
```



As a front-end developer, I  
want to request translations  
for the page I am displaying to  
the end users so that they can  
view the article in different  
languages

Size: 3

## User Story 2

```
@app.get("/wiki_translate/source_article", response_model=TranslateArticleResponse)
def translate_article(url: str = Query(None), title: str = Query(None), language: str = Query(...)):
    logging.info(f"Calling translate article endpoint for title: {title}, url: {url} and language: {language}")

    if url:
        title = extract_title_from_url(url)
        if not title:
            logging.info("Invalid Wikipedia URL provided.")
            raise HTTPException(status_code=400, detail="Invalid Wikipedia URL provided.")

    if not title:
        logging.info("Either 'url' or 'title' must be provided.")
        raise HTTPException(status_code=400, detail="Either 'url' or 'title' must be provided.")

    translated_wiki = wikipediaapi.Wikipedia(user_agent='MyApp/1.0 (contact@example.com)', language=language)
    translated_page = translated_wiki.page(title)

    if not translated_page.exists():
        logging.info("Translated article not found.")
        raise HTTPException(status_code=404, detail="Translated article not found.")

    translated_content = translated_page.text if translated_page.text else ""

    return {"translated_article": translated_content}
```

## User Story 3



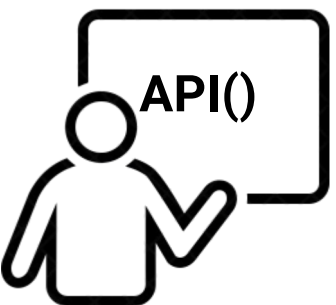
As a backend developer, I want to configure middleware for API access, so that I can ensure secure access to data.

Size 8



```
31     # Allow all origins (be cautious with this in production)
32     app.add_middleware(
33         CORSMiddleware,
34         allow_origins=["*"], # You can specify the allowed origins here
35         allow_credentials=True,
36         allow_methods=["*"],
37         allow_headers=["*"],
38     )
```

<Demo/>



## What Worked



01

---

Correspondence  
& Collaboration



02

---

Research  
Segmentation

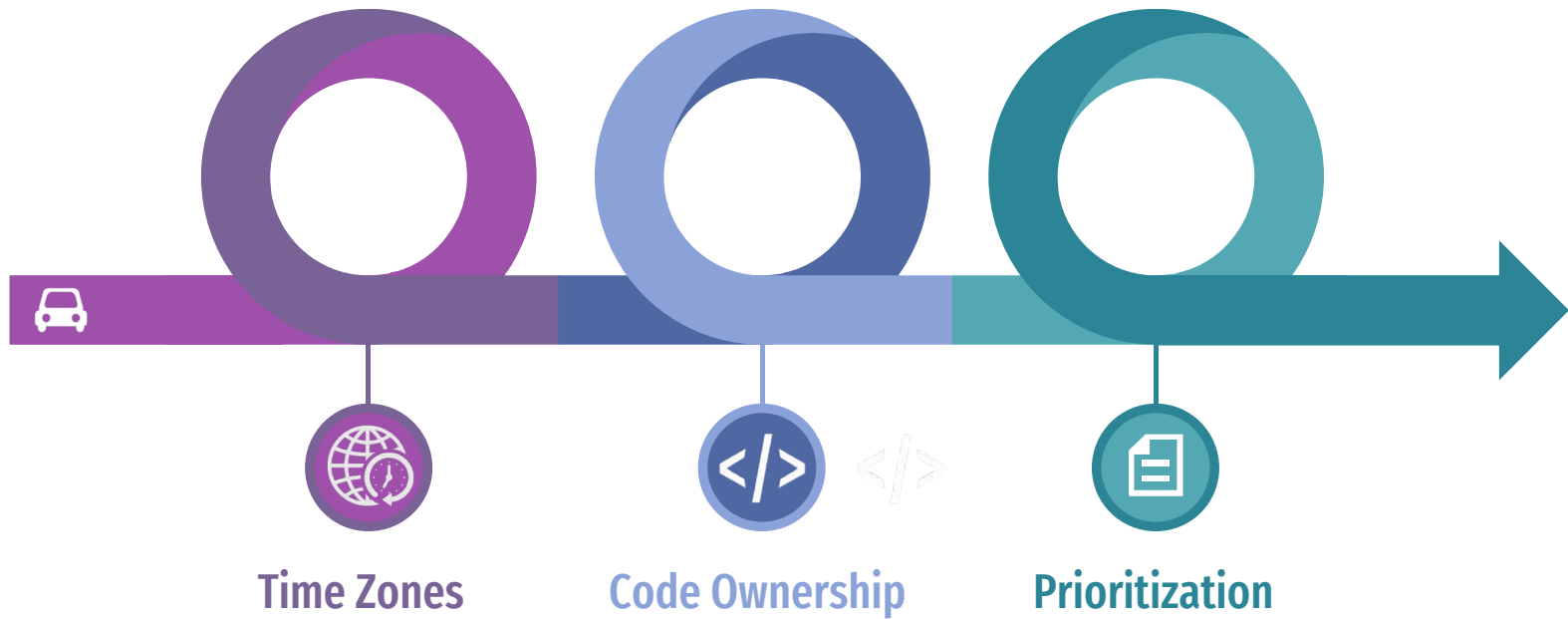


03

---

Mob  
Configuration

# Roadblocks



# Lessons Learned

## Intergroup Development

Coding with other teams

## Regional Time Clarity

Meeting confirmation



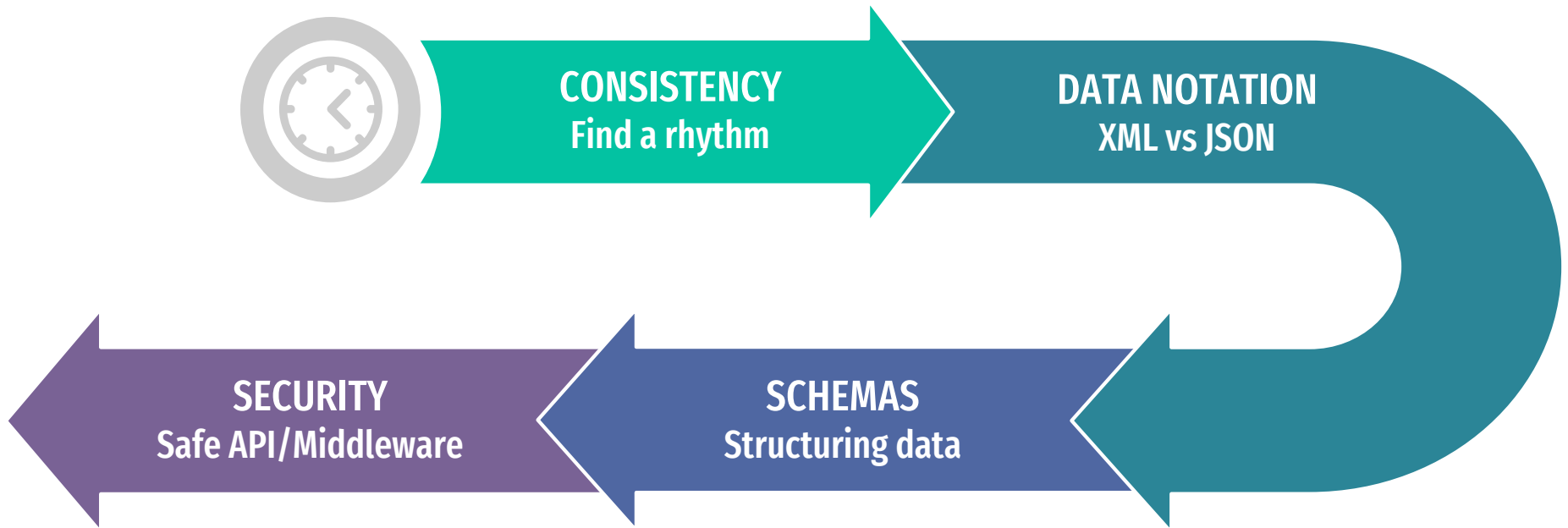
## Accountability

Last minute work

## Documentation

Finding solutions

## Future



# Thank you



**The problem with Wikipedia is that it only works in practice. In theory, it's a total disaster.**

**- anonymous**