

Inclusion – Milestone 5

Andrew Rodriguez

19 Oct 2025

CST-391

Project Overview

- **Purpose:** Personal Bible study web app to manage verse-specific comments across translations.
- Backend: REST API with CRUD operations.
- **Database:** MySQL with multiple tables to support functionality.
- Frontend Milestone: React app successfully connected to backend API.
- Tools: Node.js, Express, JavaScript/TypeScript, Postman, MySQL Workbench

```
modifier_ob.
  mirror object to mirror
mirror_mod.mirror_object
 peration == "MIRROR_X":
elror_mod.use_x = True
mirror_mod.use_y = False
_operation == "MIRROR_Y"
____ror_mod.use_x = False
lrror_mod.use_y = True
 lrror_mod.use_z = False
 operation == "MIRROR_Z";
 rror_mod.use_x = False
 lrror_mod.use_y = False
 rror_mod.use_z = True
 melection at the end -add
   ob.select= 1
   er ob.select=1
   ntext.scene.objects.action
  "Selected" + str(modification
   irror ob.select = 0
  bpy.context.selected_obj
  lata.objects[one.name].sel
 int("please select exactly
  --- OPERATOR CLASSES ----
      mirror to the selected
    ject.mirror_mirror_x"
 ext.active_object is not
```

Implementation and Key Learnings

Implementation:

React components and services integrated with REST API

Routing established for Books, Verses, and Comments pages (via React Router)

Verses displayed dynamically by translation

Comments created and retrieved by verse

Key Learnings:

Define API contracts early for smoother integration

Postman testing catches backend issues before frontend work

Modular React components and reusable hooks aid debugging and reuse

Keeping schema diagrams updated reduces relation errors

Challenges and Pending Issues

- Challenges: Aligning React Router routes with backend endpoints
- Debugging asynchronous API calls and component state
- **CORS** setup and environment configuration
- **1** Pending Issues: Dropdowns for book selection incomplete
- Performance concerns with large verse datasets
- Input validation for comments still pending

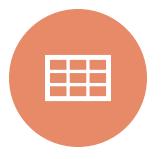
Next Steps



Update Home Page for cleaner appearance



Standardize error handling & validation across API



Optimize database queries for larger translations



Conduct broader user testing across translations