Assignment 3a

Server.js

```
const express = require("express");
const path = require("path");

const app = express();
const PORT = 3000;

// Serve static files from the 'public' folder
app.use(express.static(path.join(__dirname, "public")));

// Default route
app.get("/", (req, res) => {
  res.sendFile(path.join(__dirname, "public", "index.html"));
});

// Start the server
app.listen(PORT, () => {
  console.log(`Narrate website is running at
http://localhost:${PORT}`);
});
```

Assignment 3b

Model/Post.js

```
const mongoose = require("mongoose");

const PostSchema = new mongoose.Schema({
  title: { type: String, required: true },
  content: { type: String, required: true },
  author: { type: String, required: true },
  createdAt: { type: Date, default: Date.now },
});

module.exports = mongoose.model("Post", PostSchema);
```

postRoutes.js

```
const express = require("express");
const router = express.Router();
const Post = require("../models/Post");

// Create Post
router.post("/create", async (req, res) => {
  try {
    const newPost = new Post(req.body);
    await newPost.save();
    res.status(201).json(newPost);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Read All Posts
```

```
router.get("/posts", async (req, res) => {
  try {
    const posts = await Post.find();
    res.status(200).json(posts);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Read Single Post by ID
router.get("/post/:id", async (req, res) => {
  try {
    const post = await Post.findById(req.params.id);
    if (!post) return res.status(404).json({ message: "Post not
found" });
    res.status(200).json(post);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Update Post
router.put("/update/:id", async (req, res) => {
  try {
    const updatedPost = await
Post.findByIdAndUpdate(req.params.id, req.body, {
      new: true,
    });
    if (!updatedPost)
      return res.status(404).json({ message: "Post not found" });
    res.status(200).json(updatedPost);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

// Delete Post
router.delete("/delete/:id", async (req, res) => {
  try {
    const deletedPost = await
Post.findByIdAndDelete(req.params.id);
    if (!deletedPost)
      return res.status(404).json({ message: "Post not found" });
    res.status(200).json({ message: "Post deleted successfully"
});
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

module.exports = router;
```