Migration Module Fix Summary

Issue Description

The Azure deployment was failing with the following error:

```
ModuleNotFoundError: No module named 'scripts'
```

This occurred because the deployment script was trying to run:

```
python -m scripts.migrate
```

But Python couldn't find the `scripts` module in its module path.

Root Cause Analysis

1. **Module Import Issue**: Python's `-m` flag requires the module to be in the Python path and properly configured as a package
2. **Working Directory**: The migration was being run from `/home/site/wwwroot` but Python wasn't finding the `scripts` package
3. **PYTHONPATH Configuration**: The Azure Web App environment needed explicit PYTHONPATH configuration

Solutions Implemented

1. Primary Fix: Direct Script Path Execution

Updated the workflow to use direct script execution instead of module import:

Before:

```
python -m scripts.migrate
```

After:

```
python scripts/migrate.py
```

This approach:

- ✅ Avoids Python module path issues
- ✅ Directly executes the script file

- ✅ Works regardless of PYTHONPATH configuration
- ✅ More reliable in containerized environments

2. Fallback Fix: Enhanced PYTHONPATH Configuration

If the primary approach fails, the workflow now tries:

```
export PYTHONPATH=/home/site/wwwroot:/home/site/wwwroot:$PYTHONPATH &&
python -m scripts.migrate
```

This approach:

- ✅ Explicitly sets the PYTHONPATH to include the working directory
- ✅ Maintains compatibility with module-style imports
- ✅ Provides a reliable fallback option

3. Package Structure Verification

Confirmed that the `scripts` directory has proper package structure:

```
scripts/
├── __init__.py          # Makes it a Python package
└── migrate.py           # The migration script
```

4. Deployment Package Inclusion

Verified that the deployment workflow includes the scripts directory:

```
- name: Create deployment package
  run: |
    mkdir -p deploy
    cp -r app deploy/
    cp -r migrations deploy/
    cp -r scripts deploy/        # ✅ Scripts directory included
    cp alembic.ini deploy/
    cp requirements.txt deploy/
    cp startup.sh deploy/
```

5. Azure Web App Configuration

The workflow sets PYTHONPATH in Azure Web App environment variables:

```
az webapp config appsettings set --name ai-event-planner-saas-py --
resource-group "${{ secrets.AZURE_RESOURCE_GROUP }}" --settings \
    "PYTHONPATH=/home/site/wwwroot"  # ✅ PYTHONPATH configured
```

Migration Script Analysis

The `scripts/migrate.py` file is properly structured:

```
#!/usr/bin/env python
import os
import sys
from alembic import command
from alembic.config import Config

def run_migrations():
    """Run database migrations using Alembic."""
    print("Running database migrations...")

    # Get the directory of this script
    dir_path = os.path.dirname(os.path.realpath(__file__))

    # Create Alembic configuration
    alembic_cfg = Config(os.path.join(dir_path, "..", "alembic.ini"))

    try:
        # Run the migration
        command.upgrade(alembic_cfg, "head")
        print("Migrations completed successfully!")
    except Exception as e:
        print(f"Error running migrations: {e}")
        sys.exit(1)

def main():
    """Main entry point for the migration script."""
    run_migrations()

if __name__ == "__main__":
    main()
```

Key features:

- ✅ Proper shebang line for direct execution
- ✅ Relative path handling for `alembic.ini`
- ✅ Error handling and logging
- ✅ Exit code management for CI/CD

Testing the Fix

To test the migration script locally:

```bash
# Option 1: Direct execution
python scripts/migrate.py

# Option 2: Module execution (with PYTHONPATH)
export PYTHONPATH=.:$PYTHONPATH
python -m scripts.migrate

# Option 3: From scripts directory
cd scripts
python migrate.py
```

Workflow Changes Summary

Changed Files:

- `.github/workflows/azure-deploy-saas.yml`

Key Changes:

1. **Primary migration command**: `python scripts/migrate.py` (line ~380)
2. **Fallback migration command**: Enhanced PYTHONPATH setup (line ~395)
3. **Better error handling**: Improved logging and troubleshooting

Workflow Benefits:

- ✅ **More reliable**: Direct script execution is more predictable
- ✅ **Better error handling**: Clear error messages and fallback options
- ✅ **Improved debugging**: Enhanced logging for troubleshooting
- ✅ **Future-proof**: Works with different Python environments

Expected Results

After deploying with these changes:

1. **Primary path**: The migration should run successfully with `python scripts/migrate.py`
2. **If primary fails**: The fallback with PYTHONPATH should resolve any remaining module issues
3. **Error visibility**: Clear error messages will help diagnose any remaining issues
4. **Deployment success**: The Azure Web App should start successfully with migrated database

Monitoring and Verification

After deployment, you can verify the fix by:

1. **Check deployment logs**: Look for "Database migration completed successfully"
2. **Verify app startup**: Ensure the Azure Web App starts without errors
3. **Test application**: Confirm database connectivity and functionality

Future Recommendations

1. **Prefer direct script execution**: Use `python script.py` over `python -m module`
2. **Test locally first**: Always test migration scripts in similar environments
3. **Maintain package structure**: Keep `__init__.py` files for module compatibility
4. **Monitor deployment logs**: Watch for any Python path related issues

---

Quick Reference

Problem: `ModuleNotFoundError: No module named 'scripts'`
Solution: Use `python scripts/migrate.py` instead of `python -m scripts.migrate`
Files Changed: `.github/workflows/azure-deploy-saas.yml`