Git Commands & Deployment Guide

Summary of Changes

All security fixes have been implemented and committed to the `main` branch. There are **8 commits** ready to be pushed to the remote repository.

Commit History

```
acd1078 - chore: Add PDF to gitignore
65ce0e5 - docs: Add comprehensive implementation summary
9b084da - feat(high): Configure CORS and enhance security headers
935e831 - feat(high): Implement centralized error handling system
d0deb8b - feat(high): Add comprehensive input validation and sanitization
92e015c - feat(critical): Secure API keys with serverless functions
1cd4d64 - feat(critical): Implement Row-Level Security policies
310d1d9 - fix(critical): Fix environment variable configuration
```

View Changes

View all commits

```
cd /home/ubuntu/ai-conversation-layer
git log --oneline -8
```

View detailed changes for a specific commit

```
git show 310d1d9  # Environment variables fix
git show 1cd4d64  # RLS policies
git show 92e015c  # Serverless functions
# etc.
```

View all changed files

```
git diff origin/main --name-only
```

View complete diff

```
git diff origin/main
```

Push Changes to GitHub

Option 1: Push directly (if you have write access)

```
cd /home/ubuntu/ai-conversation-layer
git push origin main
```

Option 2: Create a feature branch and Pull Request

```
cd /home/ubuntu/ai-conversation-layer

# Create a new branch
git checkout -b security-fixes-implementation

# Push the branch
git push origin security-fixes-implementation

# Then create a Pull Request on GitHub:
# https://github.com/aiscale-coupons/AI-Conversation-Layer/compare/main...security-
fixes-implementation
```

Option 3: If you don't have push access

You'll need to:
1. Fork the repository on GitHub
2. Add your fork as a remote
3. Push to your fork
4. Create a Pull Request from your fork to the original repository

```
cd /home/ubuntu/ai-conversation-layer

# Add your fork as a remote (replace USERNAME with your GitHub username)
git remote add fork https://github.com/USERNAME/AI-Conversation-Layer.git

# Push to your fork
git push fork main

# Then create a PR on GitHub from your fork to the original repo
```

After Pushing: Deployment Steps

1. Set Environment Variables in Vercel

Go to Vercel Dashboard (https://vercel.com/dashboard) → Your Project → Settings → Environment Variables

Add the following variables:

Frontend Variables (All environments: Production, Preview, Development):
- `VITE_SUPABASE_URL` = your_supabase_project_url
- `VITE_SUPABASE_ANON_KEY` = your_supabase_anon_key

Backend Variables (Production and Preview only):
- `VITE_API_KEY` = your_gemini_api_key

OR
- `GEMINI_API_KEY` = your_gemini_api_key

2. Deploy to Vercel

If your project is connected to GitHub, Vercel will automatically deploy when you push to the main branch.

Or manually deploy:

```
npm install -g vercel
cd /home/ubuntu/ai-conversation-layer
vercel --prod
```

3. Apply Database Migrations

Option A: Using Supabase CLI (Recommended)

```
# Install Supabase CLI (if not already installed)
npm install -g supabase

# Link your project (you'll need your project reference)
cd /home/ubuntu/ai-conversation-layer
supabase link --project-ref your-project-ref

# Apply migrations
supabase db push
```

Option B: Manually via Supabase Dashboard
1. Go to Supabase Dashboard (https://app.supabase.com)
2. Navigate to SQL Editor
3. Copy and execute `supabase/migrations/20241109000001_enable_rls_and_auth.sql`
4. Copy and execute `supabase/migrations/20241109000002_create_rls_policies.sql`

Important: If you have existing data, you must assign user_id values:

```
-- Get your user ID first
SELECT id, email FROM auth.users;

-- Update existing records (replace 'USER_UUID' with actual user ID)
UPDATE campaigns SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE contacts SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE domains SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE inboxes SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE sequences SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE email_steps SET user_id = 'USER_UUID' WHERE user_id IS NULL;
UPDATE replies SET user_id = 'USER_UUID' WHERE user_id IS NULL;
```

4. Install Dependencies

Make sure to install the new dependencies:

```
cd /home/ubuntu/ai-conversation-layer
npm install
```

5. Test Locally (Optional but Recommended)

```bash
cd /home/ubuntu/ai-conversation-layer

# Install dependencies
npm install

# Create .env file from .env.example
cp .env.example .env

# Fill in your environment variables in .env
nano .env

# Start development server
npm run dev

# Test serverless functions locally (requires Vercel CLI)
vercel dev
```

Testing Checklist

After deployment, test the following:

- [ ] Frontend loads without errors
- [ ] Environment variables are working (check browser console)
- [ ] User authentication works
- [ ] RLS is enforced (users can only see their own data)
- [ ] API endpoints respond correctly (`/api/generate-email` , `/api/detect-intent`)
- [ ] Rate limiting works (make 25+ requests quickly)
- [ ] Input validation rejects invalid data
- [ ] Error messages are user-friendly (no sensitive info exposed)
- [ ] CORS headers are present (check Network tab in DevTools)
- [ ] Security headers are present (use https://securityheaders.com)

Rollback Plan

If something goes wrong after deployment:

Rollback Code Changes

```bash
cd /home/ubuntu/ai-conversation-layer

# View recent commits
git log --oneline

# Rollback to a specific commit (replace COMMIT_HASH)
git reset --hard COMMIT_HASH

# Force push to remote (⚠️ use with caution)
git push origin main --force
```

Rollback Database Migrations

Run the rollback SQL in Supabase Dashboard:

```sql
-- Drop all RLS policies
DROP POLICY IF EXISTS "Users can view their own campaigns" ON campaigns;
DROP POLICY IF EXISTS "Users can insert their own campaigns" ON campaigns;
-- ... (repeat for all policies)

-- Disable RLS
ALTER TABLE campaigns DISABLE ROW LEVEL SECURITY;
ALTER TABLE contacts DISABLE ROW LEVEL SECURITY;
-- ... (repeat for all tables)

-- Remove user_id columns (optional, will lose data association)
ALTER TABLE campaigns DROP COLUMN IF EXISTS user_id;
-- ... (repeat for all tables)
```

Troubleshooting

Issue: Push rejected (no write access)

```bash
# Check your remote URL
git remote -v

# If using HTTPS, you may need to authenticate
# Consider using SSH instead:
git remote set-url origin git@github.com:aiscale-coupons/AI-Conversation-Layer.git
```

Issue: Merge conflicts

```bash
# Pull latest changes first
git pull origin main

# Resolve conflicts in your editor
# Then commit the merge
git add .
git commit -m "Merge main and resolve conflicts"
git push origin main
```

Issue: Environment variables not working in Vercel

- Ensure variables are set for the correct environment (Production/Preview)
- Redeploy after setting environment variables
- Check spelling of variable names

Issue: Database migrations failed

- Check Supabase logs for specific errors
- Ensure you have proper permissions
- Verify that table names match your schema

Next Steps After Deployment

1. **Monitor Errors**: Check Vercel logs for any runtime errors
2. **Test Thoroughly**: Go through all features to ensure they work
3. **Update Documentation**: Update README.md with any deployment notes

4. **Set Up Monitoring**: Consider integrating Sentry or LogRocket
5. **Performance Testing**: Test with realistic data volumes
6. **Security Audit**: Run a security scan on your deployed app

Additional Resources

- IMPLEMENTATION_SUMMARY.md (./IMPLEMENTATION_SUMMARY.md) - Complete implementation details
- Vercel Deployment Docs (https://vercel.com/docs/deployments/overview)
- Supabase Migration Docs (https://supabase.com/docs/guides/cli/local-development#database-migrations)
- RLS Documentation (https://supabase.com/docs/guides/auth/row-level-security)

Ready to deploy? Start with:

```
cd /home/ubuntu/ai-conversation-layer
git push origin main
```

Then follow the deployment steps above! 🚀