# Technical Installation Manual: The Republic Project

## Table of Contents


1. Introduction
2. Prerequisites
3. Repository Setup
4. Environment Configuration
5. Supabase Configuration
6. External API Configuration
7. Deployment and Running
8. Troubleshooting
9. Additional Resources


## Introduction

This comprehensive manual provides step-by-step instructions for setting up and deploying The Republic project, which consists of a Next.js frontend, python flask load balancer and an Express backend within a single repository.

## Prerequisites

Before beginning the installation process, ensure you have the following software installed:

### Software Requirements

- Python (3.10.12 or later)
- Pip (24.2 or later)
- Node.js (v18.0.0 or later)
- Git
- npm or yarn (package managers)

PROF

### Installation Resources

- Python: https://www.python.org/
- Pip: https://pypi.org/project/pip/
- Node.js: https://nodejs.org/
- Git: https://git-scm.com/downloads
- npm Documentation: https://docs.npmjs.com/
- Yarn Installation: https://classic.yarnpkg.com/en/docs/install/

## Repository Setup

### Cloning the Repository

Open your terminal and run the following commands:

```
git clone https://github.com/COS301-SE-2024/The-Republic
cd The-Republic
```

*Note: If you've downloaded the project as a .zip file, extract it and navigate to the project directory instead.*

## Installing Dependencies

### Frontend (Next.js)

From the root directory, navigate to the `frontend` directory and install dependencies:

```
cd frontend
npm install
```

### Backend (Express)

From the root directory, navigate to the `backend` directory and install dependencies:

```
cd backend
npm install
```

### Load Balancer (Python)

From the root directory, navigate to the `proxy/python` directory and install dependencies:

Start by creating a virtual environment

PROF

```
cd /proxy/python
sudo apt-get update
sudo apt-get install python3-venv
python3 -m venv venv

pip install virtualenv
virtualenv -p python3 <env_name>

# activating virtualenv
source <env_name>/bin/activate
# deactivating virtualenv
deactivate
```

Install dependencies

```
```

```
cd proxy/python
pip install -r requirements.txt
```

## Environment Configuration

### Frontend (.env file)

Create a `.env` file in the `frontend` directory with the following variables:

```
NEXT_PUBLIC_SUPABASE_URL=<Your Supabase URL>
NEXT_PUBLIC_SUPABASE_ANON_KEY=<Your Supabase Anon Key>
NEXT_PUBLIC_BACKEND_URL=http://localhost:8080
NEXT_PUBLIC_GOOGLE_MAPS_API_KEY=<Your Google Maps API Key>
NEXT_PUBLIC_AZURE_CONTENT_MODERATOR_URL=<Azure Content Moderator URL>
NEXT_PUBLIC_AZURE_CONTENT_MODERATOR_KEY=<Azure Content Moderator Key>
NEXT_PUBLIC_AZURE_IMAGE_CONTENT_SAFETY_URL=<Azure Image Content Safety
URL>
NEXT_PUBLIC_AZURE_IMAGE_CONTENT_SAFETY_KEY=<Azure Image Content Safety
Key>
NEXT_PUBLIC_FRONTEND_URL=http://localhost:3000
```

### Backend (.env file)

Create a `.env` file in the `backend` directory with the following variables:

```
SUPABASE_URL=<Your Supabase URL>
SUPABASE_SERVICE_ROLE_KEY=<Your Supabase Service Role Key>
SUPABASE_ANON_KEY=<Your Supabase Anon Key>
OPENAI_API_KEY=<Your OpenAI API Key>
ALLOWED_ORIGIN=<Alowed Origin, URL to Frontend app>
REDIS_URL=<Redis URL Config>
RESEND_API_KEY=<Key for Vercel's Email Send>
PORT=<Port for Runnning the Server>
```

PROF

### Load Balancer (.env file)

# Place this in (/proxy/python/.env) File in Root Directory

```
PORT=5000
MAX_RETRIES=<Maximum retries for Failed Requests>
FRONTEND_URL=<Frontend Url, Ussually http://localhost:3000>
```

```
SERVER_1=<Link to Backend Server No. 1>
SERVER_2=<Link to Backend Server No. 2>
SERVER_3=<Link to Backend Server No. 3>
SERVER_4=<Link to Backend Server No. 4>
```

# Supabase Configuration

## Account Creation

1. Visit https://supabase.com/ and sign up for an account if you don't have one.
2. Log in to your Supabase account.

## Project Creation

1. Click on "New Project" in the Supabase dashboard.
2. Enter a project name.
3. Choose a secure database password.
4. Select an appropriate region for your project.
5. Click "Create New Project" to finalize.

## Retrieving Credentials

1. Navigate to your project dashboard.
2. Go to "Settings" -> "API" to find:
    - `SUPABASE_URL`
    - `SUPABASE_ANON_KEY`
3. Go to "Settings" -> "Database" to find:
    - `SUPABASE_SERVICE_ROLE_KEY`

## Database Schema Setup

1. In the Supabase dashboard, navigate to the "SQL Editor".
2. Run the provided schema SQL or manually create the necessary tables and relationships as defined in your project requirements.

PROF

# External API Configuration

## Google Maps API

1. Go to the Google Cloud Console: https://console.cloud.google.com/
2. Create a new project or select an existing one.
3. Enable the Google Maps JavaScript API for your project.
4. Create an API key and restrict it as needed.
5. Copy the API key to your frontend `.env` file.

## Azure Content Moderator

1. Sign in to the Azure portal: https://portal.azure.com/

2. Create a new Content Moderator resource.
3. Once created, go to the "Keys and Endpoint" section.
4. Copy the endpoint URL and one of the keys to your frontend `.env` file.

### OpenAI API

1. Sign up for an OpenAI account: https://openai.com/
2. Navigate to the API section and create a new API key.
3. Copy the API key to your backend `.env` file.

## Deployment and Running

### Starting the Backend Server

From the root directory, navigate to the `backend` directory and start the Express server:

```
cd backend
npm run build
npm start
```

The backend server will typically run on `http://localhost:8080`. You may need to start the backend server on multiple ports if testing everything locally so that the load balancer can utilise all these servvers.

### Starting the Load Balancer

From the root directory, navigate to the `proxy/python` directory and install dependencies:

```
cd /proxy/python
gunicorn --workers 4 wsgi:app
gunicorn --workers 4 wsgi:app --reload
```

PROF

### Starting the Frontend Application

From the root directory, navigate to the `frontend` directory, and start the Next.js application:

```
cd frontend
npm run build
npm run start
```

The frontend application will typically run on `http://localhost:3000`.

### Accessing the Application

Open your web browser and navigate to `http://localhost:3000` to access the Next.js frontend.

## Troubleshooting

- Ensure all environment variables are correctly set in both `.env` files.
- Verify that you're using the correct version of Node.js as specified in the `.nvmrc` file (if available).
- Check that all required ports (typically 3000 for frontend and 8080 for backend) are not in use by other applications.
- If you encounter any "Module not found" errors, try deleting the `node_modules` folder and running the installation step again.

## Additional Resources

For more detailed usage instructions, please refer to the User Manual available at:

User Manual