**Build and Installation Guide**

INDEX

1. Introduction 3

2. Purpose of the Document 3

3. Audience 3

4. Software Requirements 3

5. Hardware Requirements 3

6. Network Requirements 4

7. Ensure Prerequisites 4

8. Installation Steps 4

9. Configuration 6

10. Running the Application 6

11. Deployment 7

12. Testing 7

13. Troubleshooting 8

14. Maintenance and Upgrades 9

15. Conclusion 9

1. **Introduction**

The "Build and Installation Guide" serves as a comprehensive resource detailing the step-by-step process for setting up and deploying the e-Crime Reporting web application. This guide is designed to assist developers, system administrators, and users in successfully installing, configuring, and launching the application within their environment.

1. **Purpose of the Document**

The primary objective of this guide is to facilitate a seamless installation experience for users interested in deploying the e-Crime Reporting web application. It outlines the necessary prerequisites, configuration steps, and deployment procedures required to establish a fully functional instance of the application.

Developers and administrators seeking to set up the application for testing, development, or production environments will find detailed instructions, best practices, and configuration parameters essential for a successful installation process.

1. **Audience**

This guide caters to a diverse audience, including but not limited to:

* Developers involved in application setup and configuration
* System administrators responsible for deployment and maintenance
* Users interested in testing or utilizing the e-Crime Reporting application

1. **Software Requirements:**

Operating System: Compatible with Windows.

Python: Version 3.6 or higher.

Django Framework: Version 4.2.6 or compatible.

Database: SQLite.

Web Browser: Latest version of Google Chrome, Mozilla Firefox, or Safari for optimal user experience.

Additional Libraries and Packages: Ensure the installation of necessary Python packages as specified in the application's **requirements.txt** file.

1. **Hardware Requirements**:

Processor: Dual-core processor or higher.

RAM: Minimum of 4GB RAM for testing purposes; recommended 8GB or higher for production environments.

Storage: Adequate free disk space for the installation of the application and database storage.

1. **Network Requirements**:

Internet Connection: Required for initial setup, package installation, and potential updates.

Port Availability: Ensure port availability for running the application locally or hosting it on a server.

1. **Ensure Prerequisites**:

Make sure your system meets the above-mentioned hardware, software, and network requirements. Acquire the source code for the e-Crime Reporting web application from the designated repository or source.

Find the requirements.txt which has all required installations. Use “pip install -r requirements.txt” to install all of them.

1. **Installation Steps**

**8.1 Setting Up the Development Environment**

8.1.1 Clone the Repository:

- Clone the project repository from the source using Git.

```bash

git clone <repository\_url>

```

8.1.2 Create a Virtual Environment:

- Create a virtual environment to isolate the project dependencies.

```bash

python -m venv myenv # Replace 'myenv' with your preferred environment name

```

8.1.3 Activate the Virtual Environment:

- Activate the virtual environment before proceeding.

- Windows:

```bash

myenv\Scripts\activate

```

- macOS/Linux:

```bash

source myenv/bin/activate

```

8.1.4 Install Django:

- Ensure Django is installed within the virtual environment.

```bash

pip install Django==4.2.7

```

**8.2 Configuring the Database**

8.2.1. Database Setup:

- Modify the `settings.py` file to configure the database settings according to your setup. Use SQLite.

- Update the database connection details in the `settings.py` file ( I have already done it).

**8.3 . Installing Dependencies**

8.3.1. Install Project Dependencies:

- Navigate to the project directory containing the `requirements.txt` file.

- Install all project dependencies using:

```bash

pip install -r requirements.txt

```

- This command will install all the necessary packages for the project to run.

8.3.2. Run Migrations:

- Apply database migrations to set up the database schema.

```bash

python manage.py migrate

```

8.3.3. Create Superuser (Optional):

- Create an admin account to access the Django admin panel.

```bash

python manage.py createsuperuser

```

- Follow the prompts to create an admin username, email, and password.

8.4. **SSL Configuration**

8.4.1. Run with SSL:

- Run the development server with SSL enabled using OpenSSL-generated certificates.

```bash

python manage.py runsslserver --cert cert.pem --key key.pem

```

- Replace `cert.pem` and `key.pem` with the appropriate paths and filenames for your SSL certificates or use the ones that I have generated.

**9. Configuration**

**9.1. Environment Setup (Development/Production)**

Development Environment:

- Ensure the `DEBUG` setting in `settings.py` is set to `True` for debugging features.

- Use SQLite or a lightweight database for ease of development.

- **Debug must be set to True if you wish to run the server with SSL certificates**

Production Environment:

- Set `DEBUG` to `False` in `settings.py` for security reasons.

- Use a robust database like PostgreSQL or MySQL for production.

**9.2. Email Configuratio**n

Email Settings:

- Configure email settings in `settings.py` for sending email notifications.

- Update SMTP server, port, email credentials, and other necessary settings.

- Consider using environment variables for sensitive information.

- Or Use my credentials

**10. Running the Application**

**10.1. Launching the Development Server**

Development Server:

- Run the development server using the command: `python manage.py runserver`. ( Debug can be either True or False)

- For SSL support, use: `python manage.py runsslserver --cert cert.pem --key key.pem`. (Debug must be TRUE)

**10.2. Accessing the Application**

Accessing the Application:

- Once the server is running, open a web browser.

- Visit `http://localhost:8000/user\_login/` or the appropriate URL to access the application.

- Use valid credentials to log in and start using the features.

**11. Deployment**

**11.1. Deploying to Production Servers**

Deploying the Application:

- Ensure all necessary configurations are in place for the production environment.

- Transfer the codebase to the production server using secure methods like SSH or FTP.

- Install dependencies using the requirements file: `pip install -r requirements.txt`.

- Set up the appropriate server settings, ensuring security and performance optimizations.

**11.2. Configuring Server Environment Variables**

Environment Variables Setup:

- Configure any environment-specific variables required for the production server.

- Ensure sensitive information like database credentials, secret keys, and API keys are securely stored as environment variables.

**11.3. Scaling and Load Balancing (if applicable)**

Scaling for Increased Load:

- Implement scaling strategies as needed to handle increased traffic or load.

- Set up load balancing to distribute traffic efficiently across multiple servers if scaling horizontally.

**12. Testing**

**12.1. User Testing Guidelines**

\*Guidelines for User Testing:

- Prepare a comprehensive test plan covering all functionalities of the application.

- Create test scenarios that simulate real-world use cases.

- Involve a diverse set of users to gather feedback on usability and functionality.

- Document user feedback and address any reported issues or concerns promptly.

**12.2. Running Automated Tests**

Automated Testing Process:

- Utilize Django's built-in testing framework for automated testing.

- Write test cases covering various aspects of the application, including models, views, forms, and APIs.

- Run automated tests using the `manage.py` command: `python manage.py test`.

- Analyze test results and ensure all functionalities pass the automated tests.

- Craft the test cases based on the application.

**13. Troubleshooting**

**13.1. Common Installation Issues**

Issue: Dependency Installation Failure

Resolution: Verify internet connectivity and retry installation. Check for correct package names and versions.

Issue: Database Configuration Errors

Resolution: Ensure database settings (username, password, host) are correctly configured in settings.py.

Issue: SSL Certificate Errors

Resolution: Verify certificate paths and permissions. Ensure the certificates are valid and properly configured.

**13.2. Error Handling**

Error Handling Strategies:

- Monitor Django logs (if enabled) for error details.

- Implement proper logging mechanisms within the application to track errors and exceptions.

- Create custom error pages to gracefully handle user-facing errors.

- Debug errors systematically by examining stack traces and error messages.

Handling Common Errors:

- Address 404 Page Not Found errors by checking URL configurations.

- Handle 500 Internal Server errors by checking for bugs in code logic and configurations.

**14. Maintenance and Upgrades**

**14.1. Routine Maintenance Tasks**

Database Backups: Regularly back up your database to prevent data loss in case of failures.

Security Patching: Stay updated with security releases for dependencies, libraries, and the framework itself.

Log Monitoring: Monitor logs for unusual activities or errors and take necessary actions.

Resource Optimization: Periodically review and optimize resources like database queries, API calls, and server performance.

**14.2. Upgrading to New Versions**

Framework Upgrade: Follow the official framework documentation for upgrading to newer versions.

Dependency Updates: Regularly check for updates to third-party libraries and ensure compatibility with your application.

Testing after Upgrades: Conduct thorough testing after any upgrades to ensure the application's stability and functionality.

Backup before Upgrades: Always create backups of both codebase and database before performing any upgrades.

**15. Conclusion**

**15.1. Additional Resources**

Documentation: Refer to the official framework documentation for in-depth details and updates.

Community Forums: Engage with the developer community for discussions, troubleshooting, and additional tips.

Tutorials and Guides: Explore online tutorials and guides for further insights and best practices.

**15.2. Support Contact Information**

For any queries, feedback, or technical support, feel free to reach out to:

Support Email:skr.[y2409@gmail.com](mailto:y2409@gmail.com) or yeduguri@umd.edu

We appreciate your use of E-Crime Reporting and welcome any feedback or suggestions for improvements.

Thank you for choosing E-Crime Reporting!