# Project: URL Shortener

Design Document

Version 1.1

Developer: Subhajit Das

# Revision History

| Date (dd/mm/yyyy) | Version | Description | Author |
|---|---|---|---|
| 16/02/2019 | 1.0 | Design Document of URL Shorten Project. | Subhajit Das |
| 16/02/2019 | 1.1 | Mention order of execution of tests. | Subhajit Das |

# Table of Contents

# 1. Introduction of Design Document

The detailed study about different operations that are performed by a system and the relationship between them within or outside of the system is called Analysis of a system or the given project.

Objective of Design Document:

The main objective of this design is to analyse and understanding the system in detail. In this way the features and constitute parts of at least feasible solution are identified and then documented.

Road Map for Projects:

The design activity provides a roadmap to progressively transform the requirements through a number on stages into the final product by describing the structure of the system to be implemented.

Activities Used in This Design Document:

Different activities that are used in this document are briefly explained below.

Entity Relationship Diagram (ERD):

It is graphical representation of entities and their relationships to each other.

Sequence Diagrams:

It is a most popular kind Interaction Diagram. It is helpful in both when checking requirements and when checking designs.

Architecture Design Diagram:

This describes the layered Architectural model for the software system.

Class Diagram:

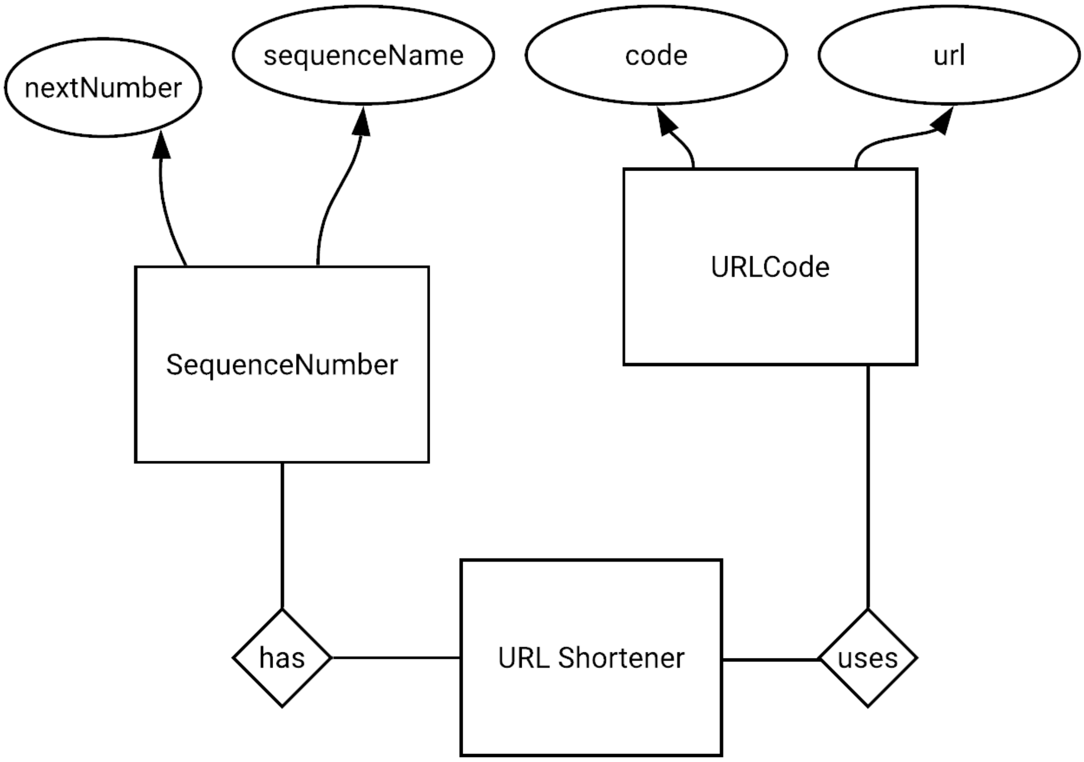
This describes the system by showing the classes, their attributes and relationships among objects etc.

Interface Design:

This design shows the important features of a product.

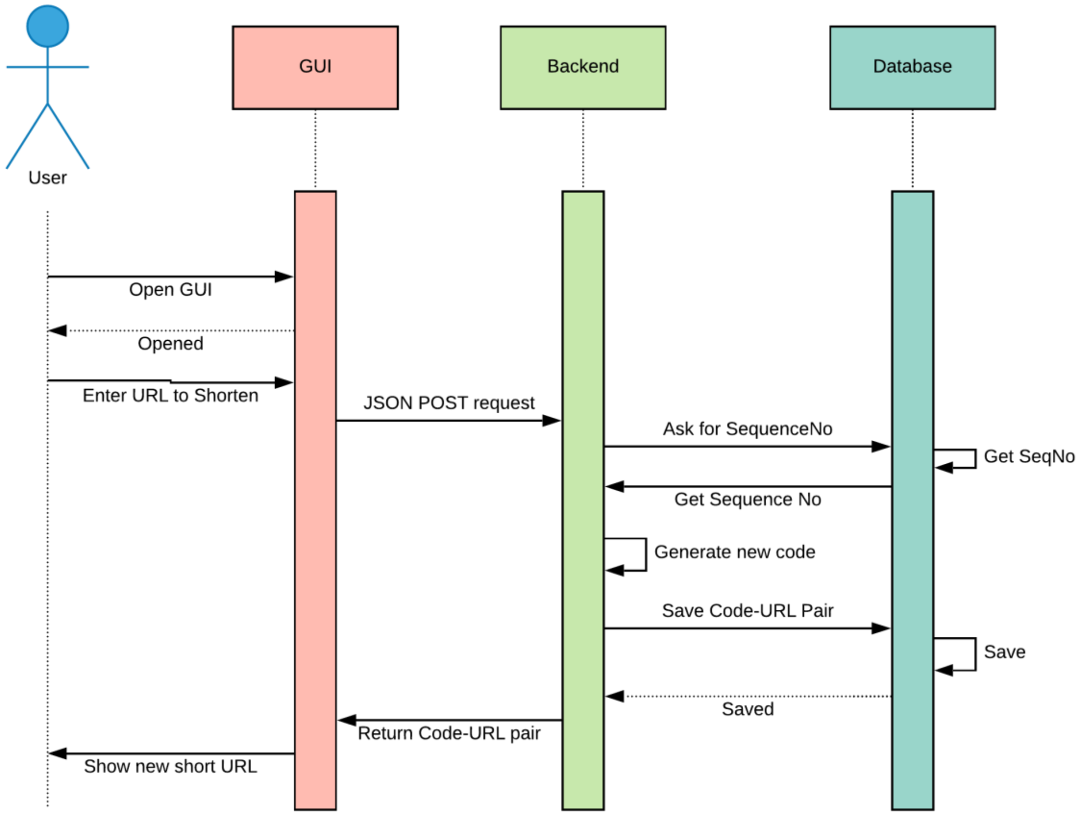
Test Cases: Finally test cases are provided in detail.

# 2. Entity Relationship Diagram (ERD)

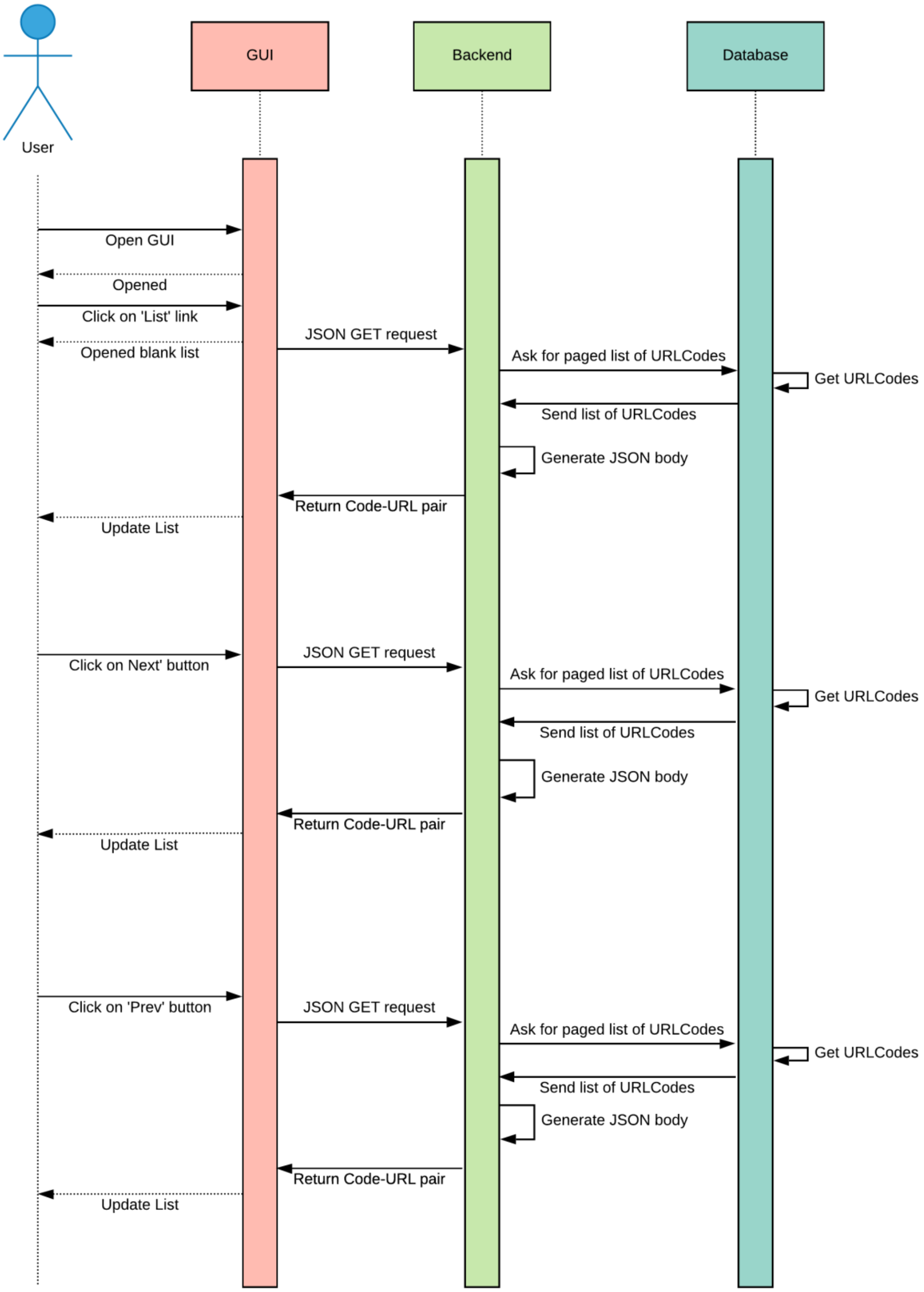


# 3. Sequence Diagrams

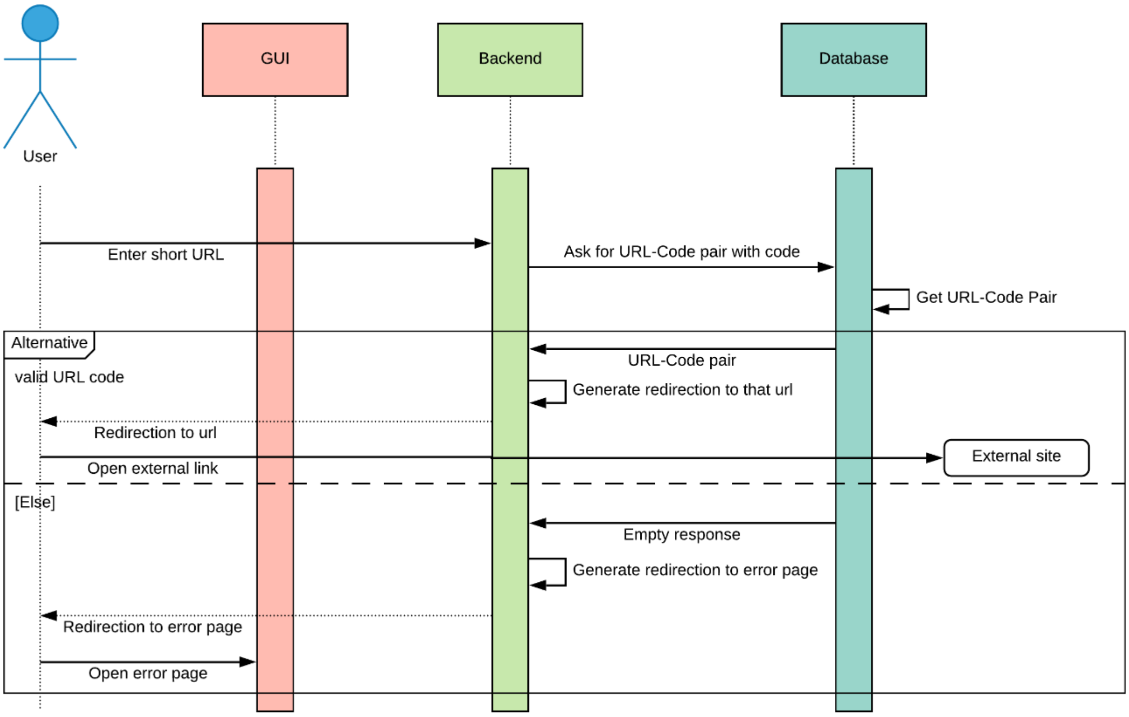
## 1. Add/Shorten URL:



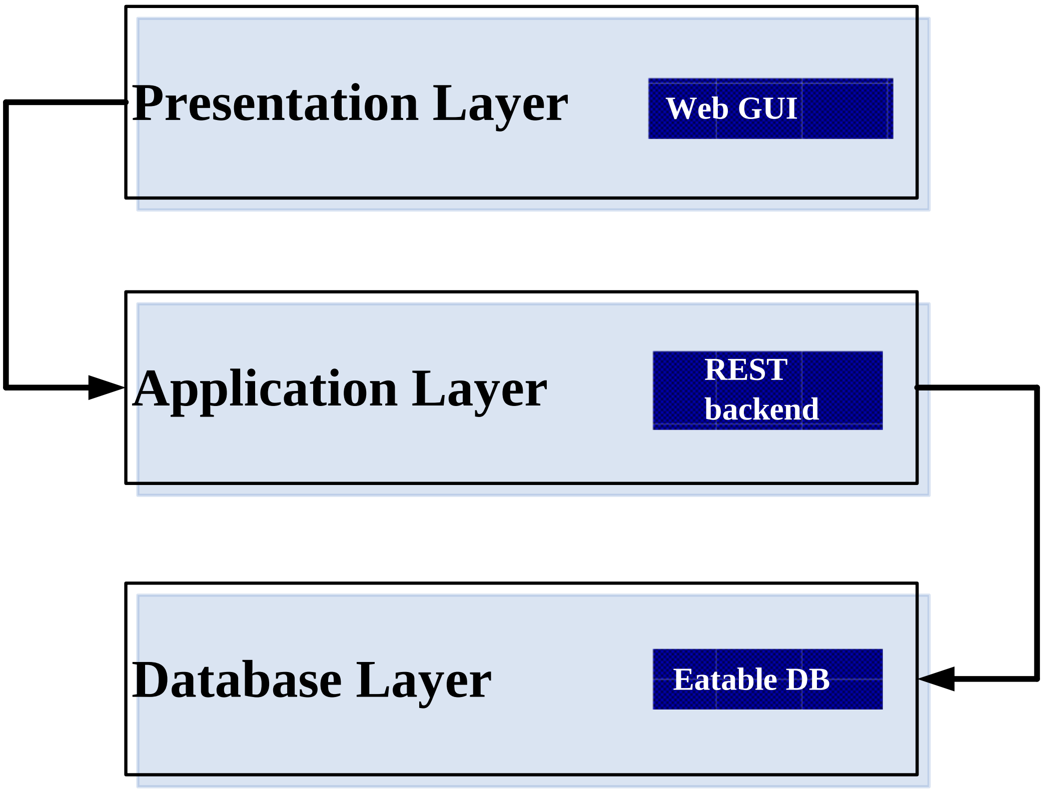
## 2. List All URLs:



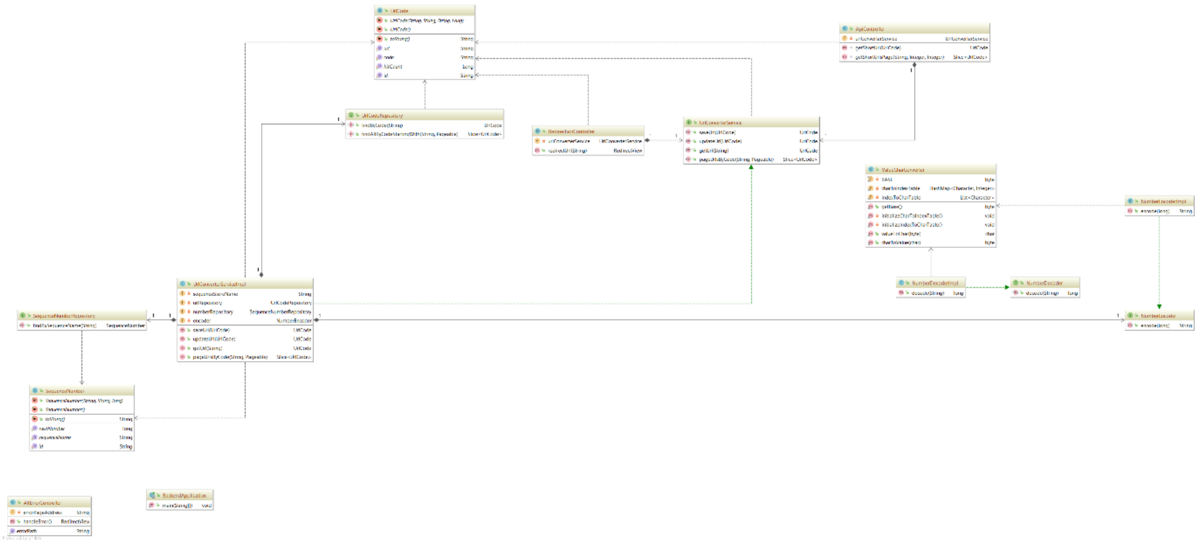
## 3. Redirection:



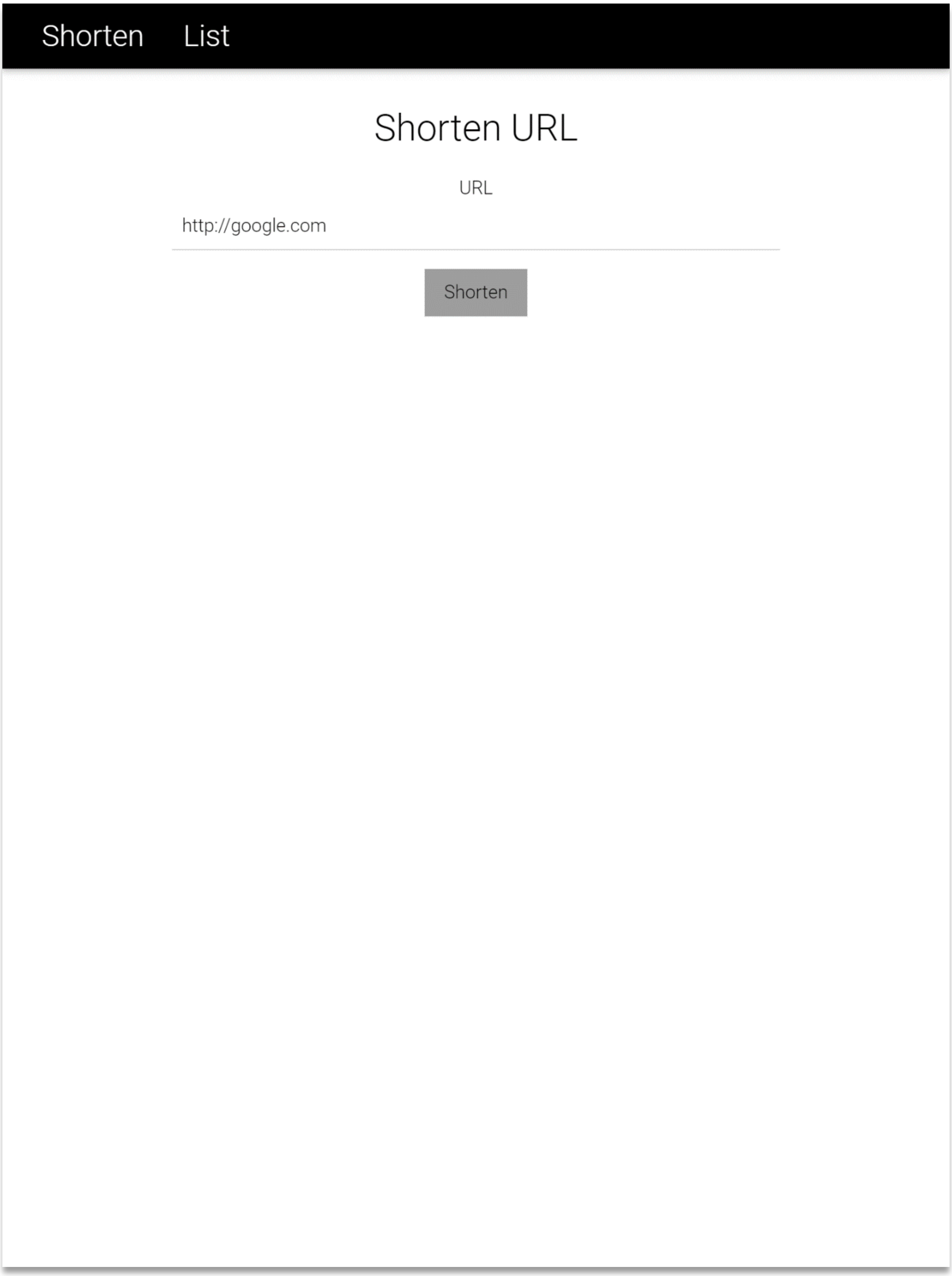
# 4. Architecture Design Diagram



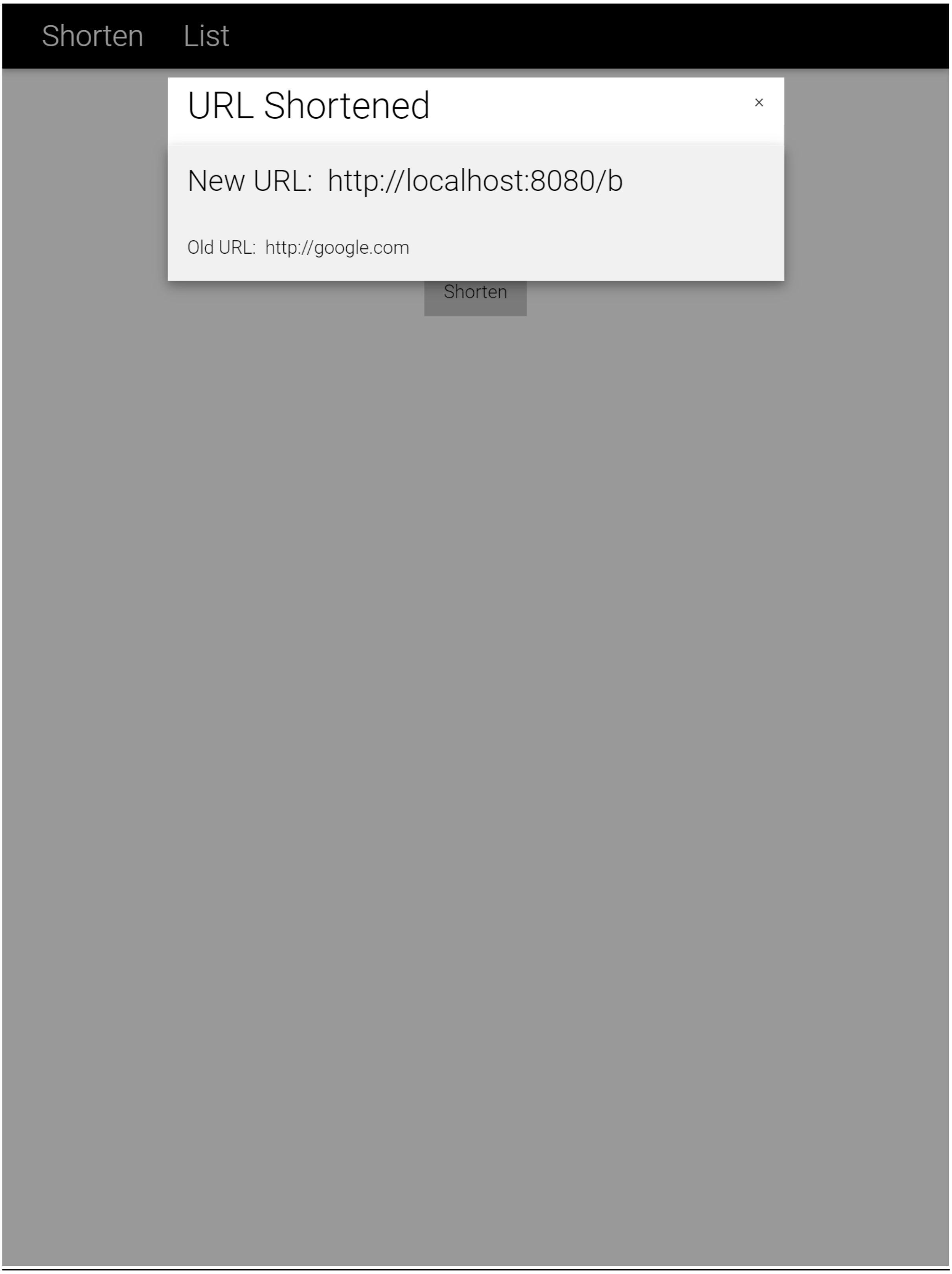
# 5. Class Diagrams


UrlCode
UrlCode(String, String, String, Long)
UrlCode()
toString() String
url String
code String
hitCount Long
id String
ApiController
urlConverterService UrlConverterService
getShortUrl(UrlCode) UrlCode
getShortUrlsPage(String, Integer, Integer) Slice<UrlCode>
UrlCodeRepository
findByCode(String) UrlCode
findAllByCodeStartingWith(String, Pageable) Slice<UrlCode>
RedirectionController
urlConverterService UrlConverterService
redirectUrl(String) RedirectView
UrlConverterService
saveUrl(UrlCode) UrlCode
updateUrl(UrlCode) UrlCode
getUrl(String) UrlCode
pageUrlsByCode(String, Pageable) Slice<UrlCode>
ValueCharConverter
BASE byte
charToIndexTable HashMap<Character, Integer>
indexToCharTable List<Character>
getBase() byte
initializeCharToIndexTable() void
initializeIndexToCharTable() void
valueToChar(byte) char
charToValue(char) byte
NumberEncoderImpl
encode(long) String
NumberDecoderImpl
decode(String) long
NumberDecoder
decode(String) long
NumberEncoder
encode(long) String
UrlConverterServiceImpl
sequenceNumberName String
urlRepository UrlCodeRepository
numberRepository SequenceNumberRepository
encoder NumberEncoder
saveUrl(UrlCode) UrlCode
updateUrl(UrlCode) UrlCode
getUrl(String) UrlCode
pageUrlsByCode(String, Pageable) Slice<UrlCode>
SequenceNumberRepository
findBySequenceName(String) SequenceNumber
SequenceNumber
SequenceNumber(String, String, long)
SequenceNumber()
toString() String
nextNumber long
sequenceName String
id String
AllErrorController
errorPageAddress String
handleError() RedirectView
errorPath String
BackendApplication
main(String[]) void
1
1
1
1
1
1
1
Powered by yFiles

# 6. Interface Design



Shorten/Add URL



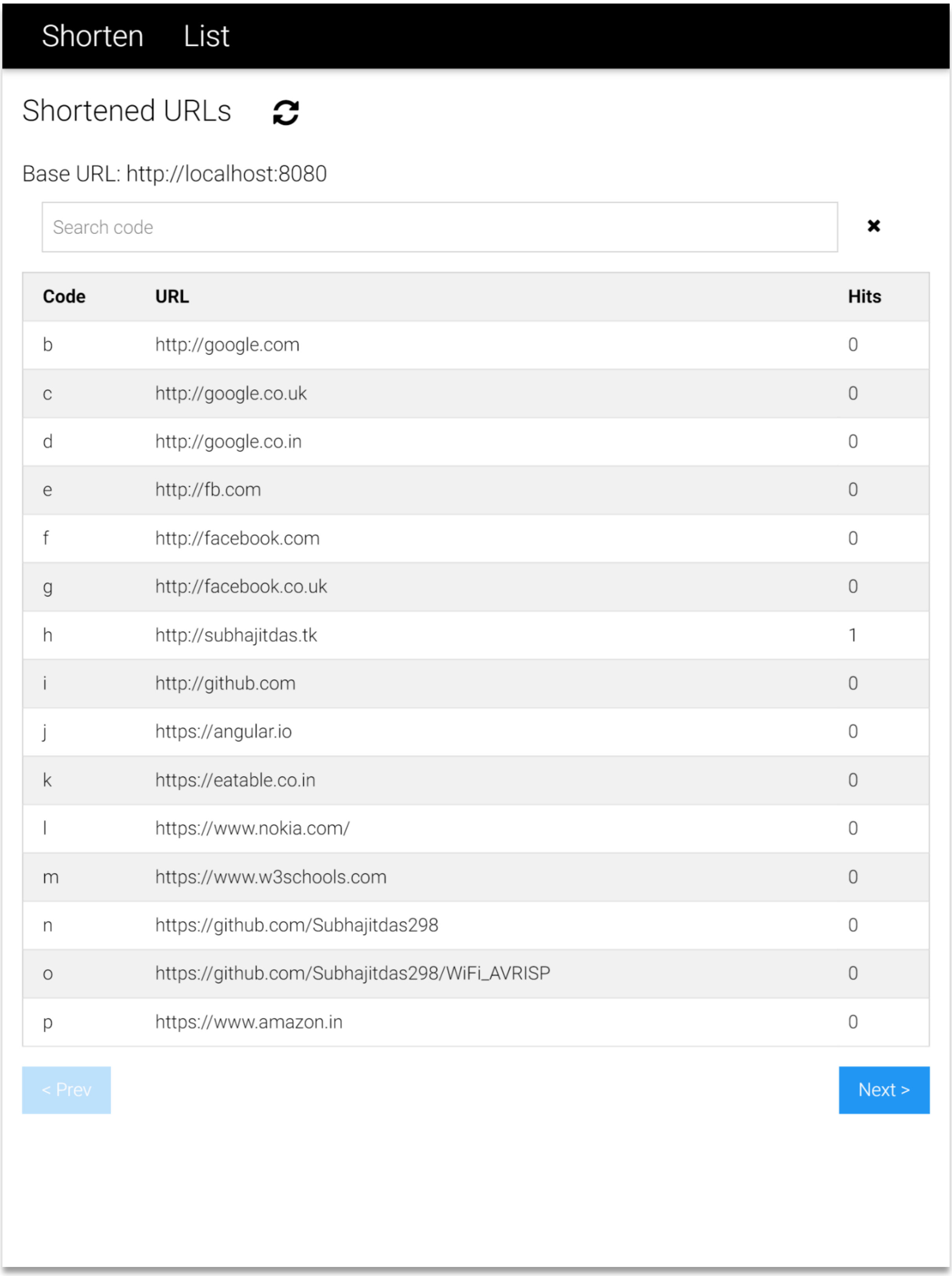
URL Shortened

New URL:  http://localhost:8080/b

Old URL:  http://google.com

Generated short URL

List URLs

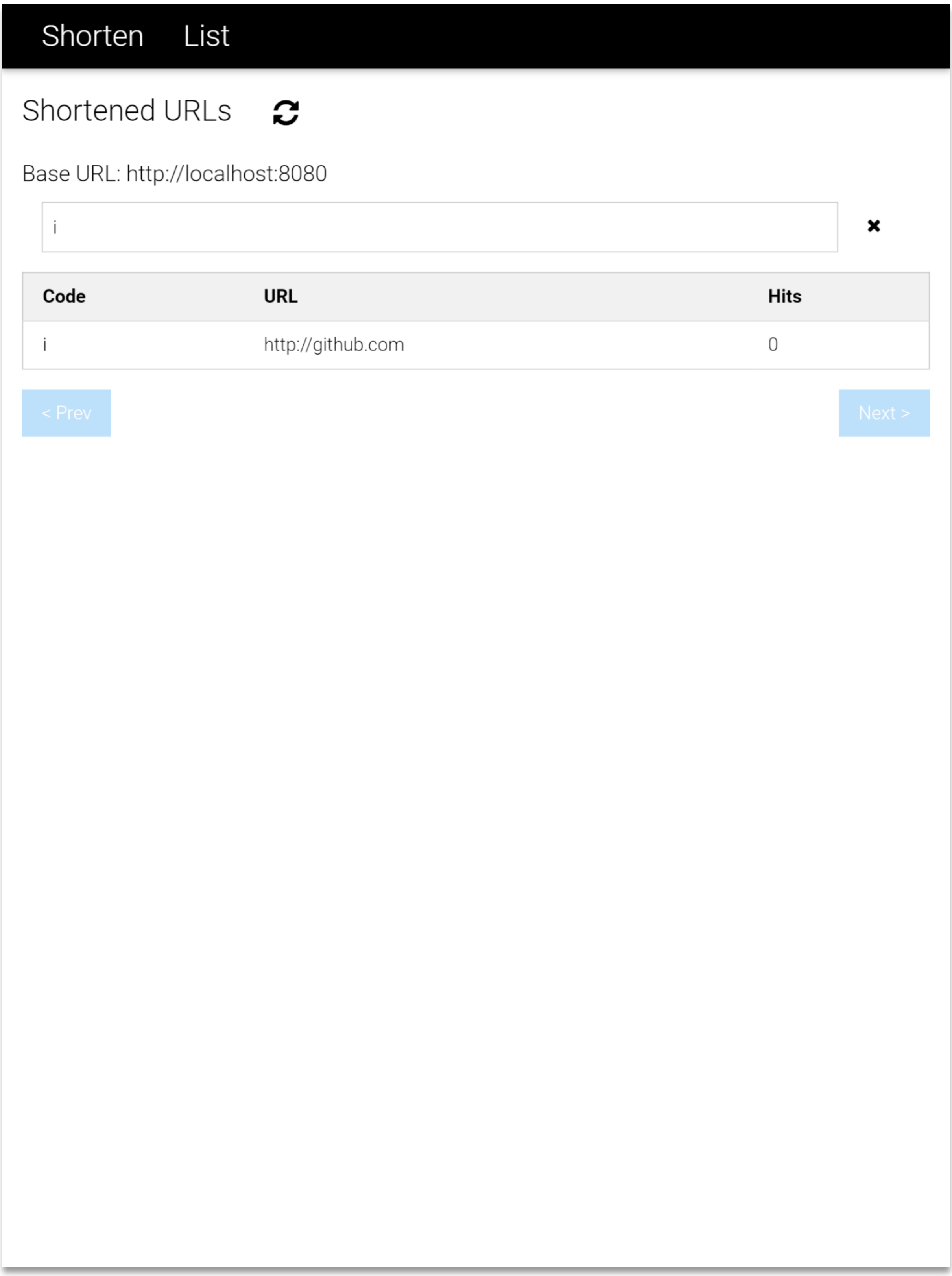


Shortened URLs

Base URL: http://localhost:8080

| Code | URL | Hits |
|---|---|---|
| b | http://google.com | 0 |
| c | http://google.co.uk | 0 |
| d | http://google.co.in | 0 |
| e | http://fb.com | 0 |
| f | http://facebook.com | 0 |
| g | http://facebook.co.uk | 0 |
| h | http://subhajitdas.tk | 1 |
| i | http://github.com | 0 |
| j | https://angular.io | 0 |
| k | https://eatable.co.in | 0 |
| l | https://www.nokia.com/ | 0 |
| m | https://www.w3schools.com | 0 |
| n | https://github.com/Subhajitdas298 | 0 |
| o | https://github.com/Subhajitdas298/WiFi_AVRISP | 0 |
| p | https://www.amazon.in | 0 |

List URLs

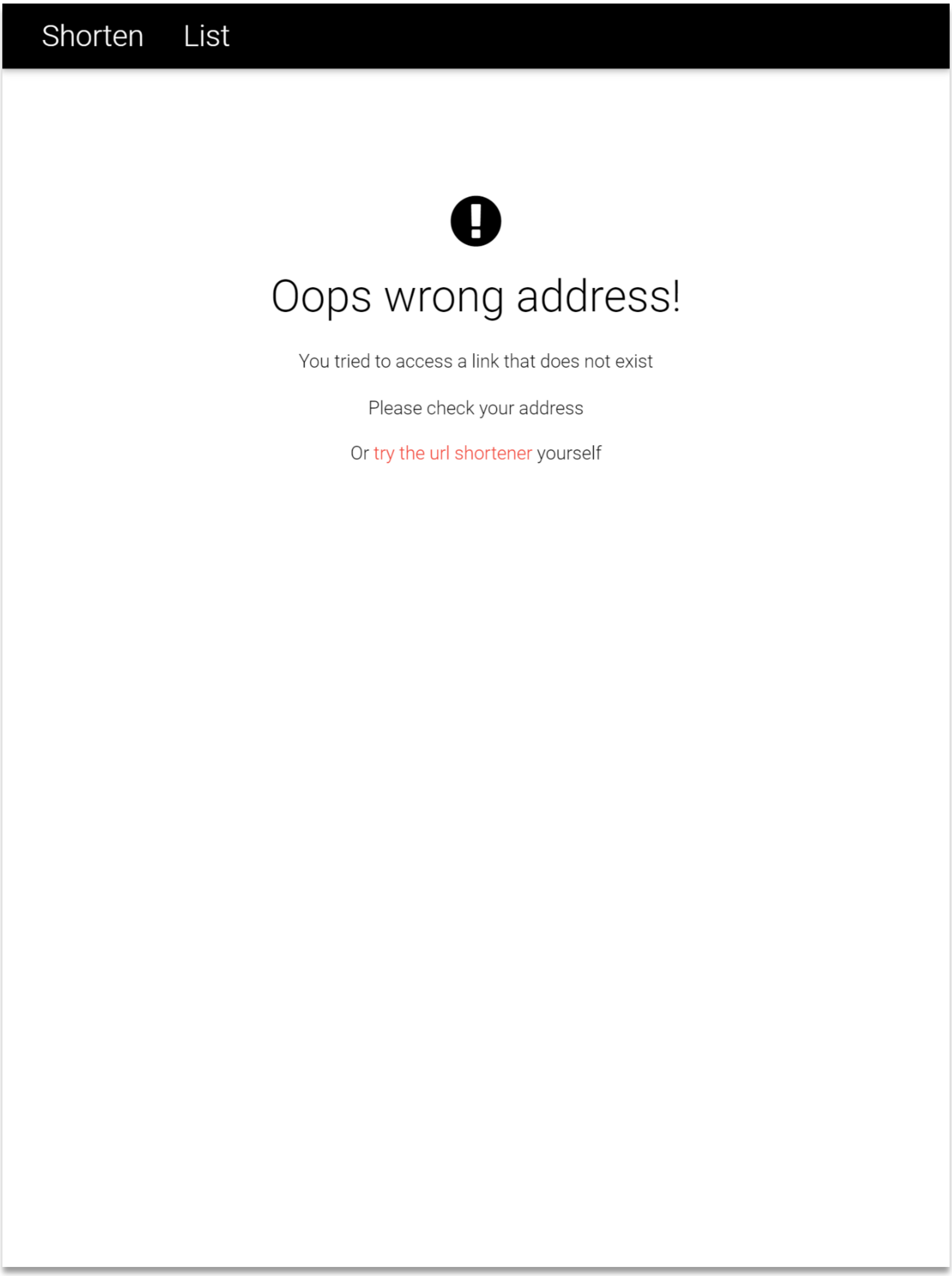


Base URL: http://localhost:8080

i

| Code | URL | Hits |
|---|---|---|
| i | http://github.com | 0 |

Search List URLs



Oops wrong address!

You tried to access a link that does not exist

Please check your address

Or try the url shortener yourself

Error screen

# 7. Test Cases

**Note:** It is to be noted that the backend and frontend have their own automated testing set up.

For backend, a fresh build or test tasks of Gradle will trigger those automated tests.

For frontend, we are required to run karma with angular test. This will run all automated tests for the front end.

To run automatic tests, the order of initialization is fixed:

Start and load database -> Run backend -> Run frontend

The test cases mentioned in this document are for manual testing of running system. This test are to be run at end.

Test Case 7.1

Title: Shorten URL

| Pre-Conditions<br>User has opened the Web UI. | | | | |
|---|---|---|---|---|
| Steps | Action | System Response Expected | Result | Tested by |
| 1. | **User** clicks "Shorten" button. | It will show error as input is less than 4 characters. | **Pass/Fail** | Subhajit Das |
| 2. | **User** fills the URL input field and press "Shorten" button. | UI will show a success message with newly generated short URL and entered URL. | **Pass/Fail** | |

Test Case 7.2

Title: List URL

| Pre-Conditions |
|---|
| • At least 11 to maximum 20 URLs are shortened.<br>• User in in shorten page. |

| Steps | Action | System Response Expected | Result | Tested by |
|---|---|---|---|---|
| 1. | User click on "List" link. | A list of all previously shortened URLs are shown. "Next" button should be active, while "Prev" button should be disabled. | Pass/Fail | Subhajit Das |
| 2. | User presses the "Next" button. | List screen will be updated with newer entries. "Next" button will be disabled. "Prev" button will be active. | Pass/Fail | |
| 3. | User presses the "Prev" button. | List screen will be updated with previous entries. "Prev" button will be disabled. "Next" button will be active. | Pass/Fail | |
| 4. | User enters search term in search box. | The list will automatically update with code items, starting with that term. | Pass/Fail | |
| 5. | User clears search box by manually or "X" button. | List restores to previously all items. | | |

Test Case 7.3

Title: Redirection

| Pre-Conditions<br>• At least 1 URLs are shortened.<br>• User must have the shortened URL. | | | | |
|---|---|---|---|---|
| Steps | Action | System Response Expected | Result | Tested by |

| | | | | |
|---|---|---|---|---|
| 1. | **User** enters a previously shortened URL into browser address bar directly. | User will be redirected to actual URL. | **Pass/Fail** | Subhajit Das |
| 2. | **User** enters an URL with invalid code, such as (0) or code which was not generated previously. | User will see the error page. | **Pass/Fail** | |