CHAPTER-1

INTRODUCTION

1.1 PROJECT OVERVIEW

Introduction to URL Shortening

URL shortening is a technique on the World Wide Web in which a Uniform Resource Locator (URL) may be made substantially shorter and still direct to the required page. This is achieved by using a redirect which links to the web page that has a long URL.

Often the redirect domain name is shorter than the original one. A friendly URL may be desired for messaging technologies that limit the number of characters in a message (for example SMS), for reducing the amount of typing required if the reader is copying a URL from a print source, for making it easier for a person to remember, or for the intention of a permalink.

Other uses of URL shortening are to "beautify" a link, track clicks, or disguise the underlying address. Although disguising of the underlying address may be desired for legitimate business or personal reasons, it is open to abuse. Some shortening services, such as goo.gl, tinyurl.com, and bit.ly can generate URLs that are human-readable.

1.2 PROJECT SPECIFICATION

PicoShare is an online URL Shortening service which aims to provide short and customizable URLs. When an original URL is provided, a unique token for the same is created. When this token is referred, the user is redirected to the original URL. Users are provided with the facility to create custom aliases or custom tokens or they can use system generated tokens.

The system has 3 types of Users:

Admin

Admin is the top-level manager of the system. Admin oversees the user management of the site and also manages the URLs in the database. All around system management under the control of the Admin.

The clients of the web application can either be registered or unregistered.

Registered Users

Registered users can Shorten URLs just like any other user. But they have an added advantage that they can reuse or update the tokens that they have once created. Users can manage their profile, Update URLs.

Guest Users

Guest uses can only use the URL shortener and no other extra functionality is available. They are restricted to use only the basic features of the website.

- Shorten URLs
- Update URLs and manage profile for registered users
- Update profile and change password for users
- Search if a token is available to use

CHAPTER-2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest Improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers.

2.2 Existing system

Some most prominent URL shortening services are goo.gl, tinyurl.com, and bit. They can generate URLs that are human-readable, although the resulting strings are longer than those generated by a length-optimized service.

These providers don’t give the facility to edit or update the URLs that were shortened by a user.

2.3 Proposed System

Registered Users can Shorten URLs. The shortened URLs can be edited or updated at a later time. They can also delete a shortened URL so that the used token is reallocated for reuse.

Guest Users can use the URL shortening service with no other extra features.

CHAPTER-3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use resources. Thus when a new project is proposed, it normally goes through a feasibility study before it's approved for development. The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and reliable in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

- **Economic Feasibility**
- **Technical Feasibility**
- **Behavioral Feasibility**

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

3.1.1 Economic Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Through the technology may become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system may still be used. So there are only minimal constraints involved with this project. The system has been developed using VB.Net, the project is technically feasible for developed.

3.1.3 Behavioral Feasibility

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 8, which is also user friendly. It does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviorally feasible.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	-	Intel Pentium IV or higher
RAM	-	2GB
Hard disk	-	512GB

3.2.2 Software Specification

• Front End	-	PHP
• Back End	-	MYSQL
• Operating System	-	Windows 10
• Technologies used	-	JS, HTML5, AJAX, JQuery, CSS
• Web Browser	-	Microsoft Internet Explorer, Mozilla Firefox, Google Chrome

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page, it now stands for PHP: Hypertext Preprocessor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page. PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP.PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

  A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

  A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard.

- **MySQL software is Open Source.**

  Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

  If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

- **MySQL Server works in client/server or embedded systems.**

  The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

  We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

CHAPTER-4

SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 USE CASE DIAGRAM

A UML use case diagram is the primary form of system/software requirements for a new software program under developed. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (such as UML). A key concept of use case modeling is that it helps us design a system from end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior. The purpose of a use case diagram in UML is to demonstrate the different ways that a user might interact with a system.

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

A use case diagram doesn't go into a lot of detail—for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.

UML is the modeling toolkit that you can use to build your diagrams. Use cases are represented with a labeled oval shape. Stick figures represent actors in the process, and the actor's participation in the system is modeled with a line between the actor and use case. To depict the system boundary, draw a box around the use case itself.

- UML use case diagrams are ideal for:
- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case
- Specify the context of a system
- Capture the requirements of a system
- Validate a systems architecture
- Drive implementation and generate test cases
- Developed by analysts together with domain experts

How to Draw a Use Case Diagram?

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analyzed, the functionalities are captured in use cases. We can say that use cases are nothing but the system functionalities written in an organized manner. The second thing which is relevant to use cases are the actors. Actors can be defined as something that interacts with the system.

Actors can be a human user, some internal applications, or may be some external applications. When we are planning to draw a use case diagram, we should have the following items identified.

- Functionalities to be represented as use case
- Actors
- Relationships among the use cases and actors.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram. The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.

- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

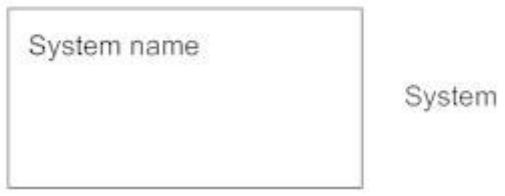
To understand the dynamics of a system, we need to use different types of diagrams. Use case diagram is one of them and its specific purpose is to gather system requirements and actors. Use case diagrams specify the events of a system and their flows. But use case diagram never describes how they are implemented. Use case diagram can be imagined as a black box where only the input, output, and the function of the black box is known. High-level design is refined again and again to get a complete and practical picture of the system. A well-structured use case also describes the pre-condition, post condition, and exceptions. These extra elements are used to make test cases when performing the testing.

Although use case is not a good candidate for forward and reverse engineering, still they are used in a slightly different way to make forward and reverse engineering. The same is true for reverse engineering. Use case diagram is used differently to make it suitable for reverse engineering. In forward engineering, use case diagrams are used to make test cases and in reverse engineering use cases are used to prepare the requirement details from the existing application.

Basic Use Case Diagram Symbols and Notations

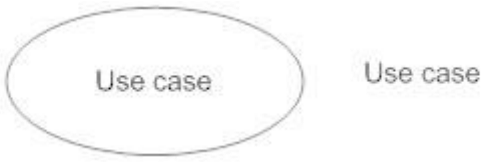
System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



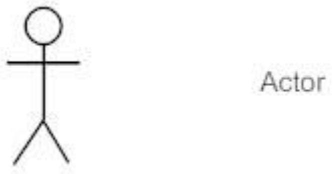
Use Case

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.



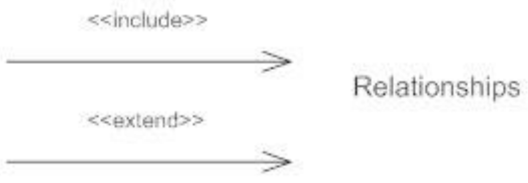
Actors

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



Relationships

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.



How to Create a Use Case Diagram

- ***Identifying Actors***

Actors are external entities that interact with your system. It can be a person, another system or an organization. In a banking system, the most obvious actor is the customer. Other actors can be bank employee or cashier depending on the role you're trying to show in the use case.

- ***Identifying Use Cases***

A good way to do this is to identify what the actors need from the system. In a banking system, a customer will need to open accounts, deposit and withdraw funds, request check books and similar functions. So, all of these can be considered as use cases. Top level use cases should always provide a complete function required by an actor. You can extend or include use cases depending on the complexity of the system. Once you identify the actors and the top-level use case you have a basic idea of the system. Now you can fine tune it and add extra layers of detail to it.

- ***Look for Common Functionality to use Include***

Look for common functionality that can be reused across the system. If you find two or more use cases that share common functionality you can extract the common functions and add it to a separate use case. Then you can connect it via the include relationship to show that it's always called when the original use case is executed.

- ***Is it Possible to Generalize Actors and Use Cases?***

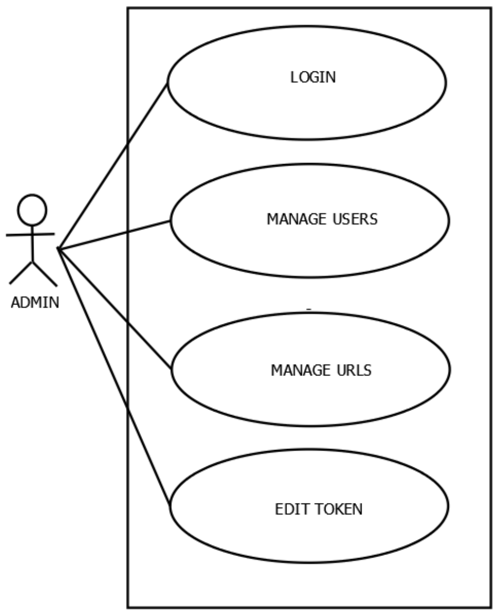
There may be instances where actors are associated with similar use cases while triggering few use cases unique only to them. In such instances, you can generalize the actor to show the inheritance of functions. You can do a similar thing for use case as well.

- ***Optional Functions or Additional Functions***

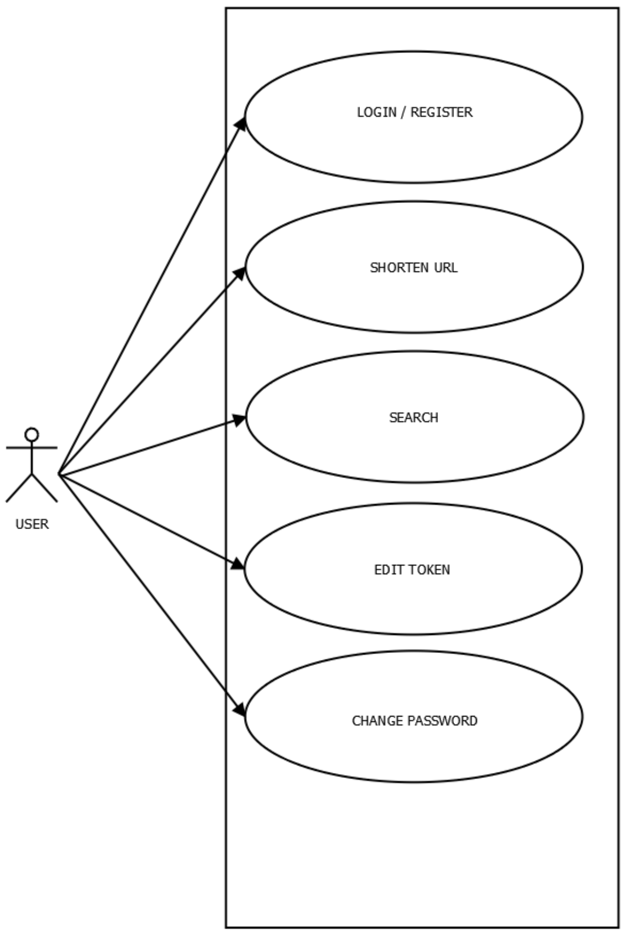
There are some functions that are triggered optionally. In such cases, you can use the extend relationship and attach an extension rule to it. In the below banking system example "Calculate Bonus" is optional and only triggers when a certain condition is matched.

Use case diagrams:

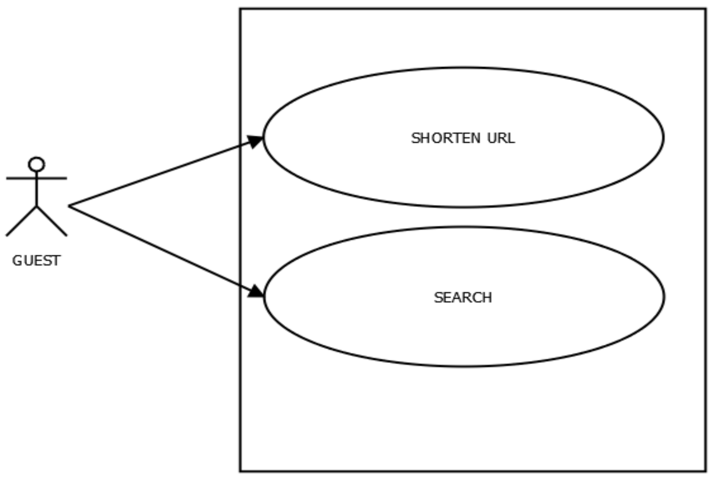
Admin:



User:


LOGIN / REGISTER
SHORTEN URL
SEARCH
EDIT TOKEN
CHANGE PASSWORD
USER

Guest:


SHORTEN URL
SEARCH
GUEST

4.3. Database Design

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

4.3.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

4.3.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.

- ✓ Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by
1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

TABLE DESIGN

1. **Table design:** tbl_login

   Primary Key: lid

Field name	Data type	Description
Lid*	int	Login id (Primary key)
regid	int	Registration id
password	Varchar(100)	Password
utype	int	User type
status	int	Status: blocked / unblocked

2. **Table design:** tbl_registration

   Primary Key: rid

Field name	Data type	Description
Rid*	Int	Registration ID (foreign key in tbl_login)
fname	Varchar(100)	First name
lname	Varchar(100)	Last name
email	Varchar(100)	Email
username	Varchar(100)	Username

mobile	Varchar(12)	Mobile no.
photo	Varchar(100)	Pro tbl_moviedetailes

3. **Table design:** tbl_shorturl

   Primary Key: urlid

Field name	**Data type**	**Description**
Urlid*	Int	URL id (Primary key)
Uid	int	User id (foreign key)
Org_url	Varchar(100)	Orginal URL
Token_url	Varchar(500)	Unique token for urls
timestamp	Varchar(500)	Cover photo

CHAPTER-5

TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency

- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop.

After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

CHAPTER-6

IMPLEMENTATION

6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.
- Training of the staff in the changeover phase.

6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won’t take place.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

CHAPTER 7

CONCLUSION AND FUTURE SCOOPE

7.1 CONCLUSION

URLs therefore have a big role to play in this digital story. They are the connectors of the web. The tool we use to communicate with online. URLs have come a long way since their inception over 20 years ago. Especially with the creation of the URL Shortener, which has come to play an important part in retaining the importance and usefulness of the URL in our ever-evolving digital landscape.URL Shorteners have moved from simply being a tool to shorten your link, to an integral tracking tool used to refine marketing initiatives and drive brand growth.

URL Shortening is one of the most popular upcoming Internet based service. As the acceptance and use of world wide web and internet is growing day by day, URL shortening servicers have a great future scope. Anyone who use the internet services can greatly benefit from URL shortening services in several ways. There are several reasons for using URL shortners in the internet world.

- Everyone prefers clean and short links
- Free URL customization
- make links more manageable
- They can be transformed into social media services
- promote sharing
- Cut marketing costs by cutting number of characters
- can provide users useful features
- Suitable for SEOs

7.2 FUTURE SCOPE

- Introducing PicoBin which is an online text storage site is a type of online content hosting service where users can store plain text.
- Improving security of PicoShare with addition of google api: Google safe search API

CHAPTER 8

BIBLIOGRAPHY

BOOKS/REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “So*ftware engineering*: a precise approach”, 2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.jquery.com
- http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
- www.agilemodeling.com/artifacts/useCaseDiagram.htm

CHAPTER 9

APPENDIX

9.1 Sample Code

Index.php

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

  <head>

    <meta charset="utf-8">

    <title></title>

  </head>

  <body style="background-image: url('images/redir.gif')">



  </body>

</html>

 <?php



    function getCurrentUri()

    {

        $basepath = implode('/', array_slice(explode('/', $_SERVER['SCRIPT_NAME']), 0, -1)) .
'/';



        $uri = substr($_SERVER['REQUEST_URI'], strlen($basepath));



        if (strstr($uri, '?')) $uri = substr($uri, 0, strpos($uri, '?'));
        $uri = '/' . trim($uri, '/');

        return $uri;

    }



    $base_url = getCurrentUri();



    $routes = array();

    $routes = explode('/', $base_url);



    // print_r($routes);

    foreach($routes as $route)

    {

        if(trim($route) != '')

            array_push($routes, $route);

    }




if($base_url == "/"){

            header('location:home.php');

}

    else if($base_url == "login.php"){

        header('location:login.php');

    }
        else if($base_url == "home.php"){

        header('location:home.php');

    }

        else if($base_url == "signup.php"){

        header('location:signup.php');

    }

else{

        require 'redirect.php';

        $token = implode("", explode('/', $base_url));

        $redirect = new redirect();

        $res =  $redirect->url_redir_($token);



        if($res){

          // echo strncasecmp($res, 'driv', 4);

            if(strncasecmp($res, 'http',4) != 0)

            {

                $res = 'http://' .$res;

            }

            // echo $res;

            header('location:'.$res);

        }else{

        header('location:error.php');

        }
```

```
   }


?>
```

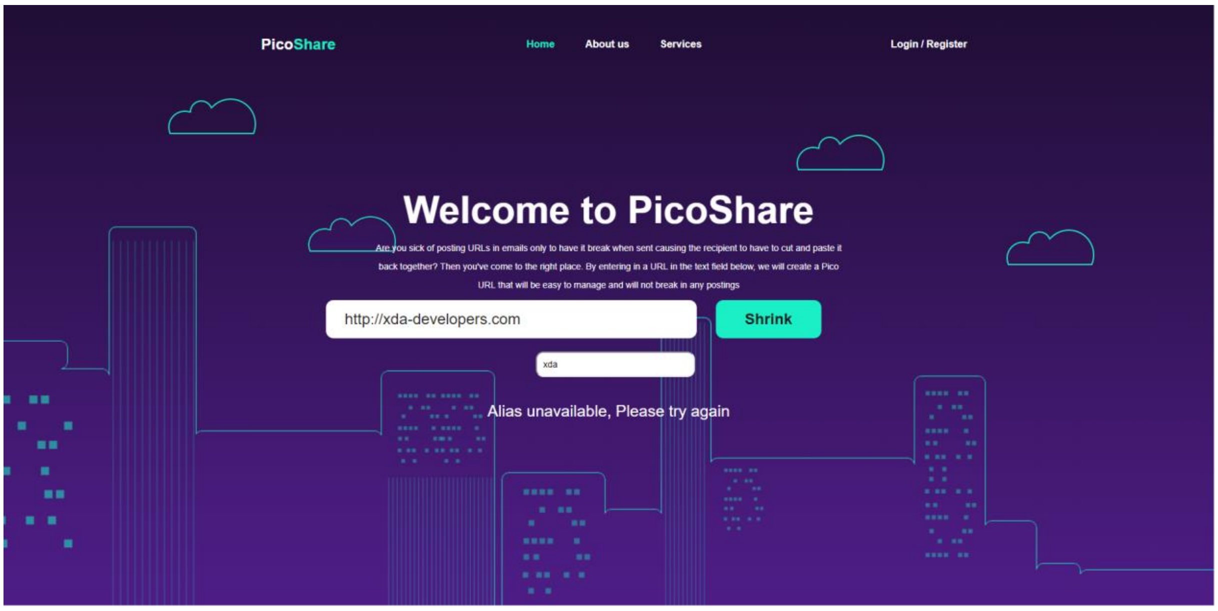
Shorter.php

```
<?php

require ("session.php");

include("connect.php");

if(isset($_POST['url']))

{

  $input_url=$_POST["url"];

  $alt=$_POST["alt"];

  $uid=getSession('uid');

  if(!$uid){

    $uid = '2';

  }

  if(isset($_POST['alt']) && $_POST['alt']!='err')

  {

    //check if alt exists//

    $qry="SELECT `tocken_url` FROM `tbl_shorturl` WHERE tocken_url='$alt'";
```

```
   $res=mysqli_query($conn,$qry) or die("error while checking tocken");

   if(mysqli_num_rows($res)==0)

   {

     //if exists

     $short_url = $alt;

   }

   else

   {

     $short_url = false;

     echo('<p style="font-size:25px; color:white; margin-top:30px">Alias unavailable, Please
try again</p>');

   }

 }

 else

 {

   //shorten url function

   $short_url_t=substr(md5($input_url.mt_rand()),0,7);

   $qry="SELECT `tocken_url` FROM `tbl_shorturl` WHERE tocken_url='$short_url_t'";

   $res=mysqli_query($conn,$qry) or die("error while checking tocken");
```

```
  if(mysqli_num_rows($res)==0)

  {

    $short_url = $short_url_t;

  }

  else

  {

    //return error

echo('<p style="font-size:25px;color:white; margin-top:30px">Tocken exists, Please try again</p>');

  }

 }

 if($short_url!=false){

 $qry="INSERT INTO `tbl_shorturl`(`uid`, `org_url`, `tocken_url`) VALUES ('$uid','$input_url','$short_url')";

  $res = mysqli_query($conn,$qry) or die("SQL error while inserting to tbl_shorturl");;

  echo '<p style="font-size:25px;color:white;  margin-top:30px">Your New URL Is:</p><br>

  <p style="font-size:10px;color:white; margin-top:-40px"><a target="_blank" class="display-4 d-block" href="http://localhost/PicoShare/'.$short_url.'">http://localhost/PicoShare/'.$short_url.'</a></p>';

 } }
```

9.2 ScreenShots



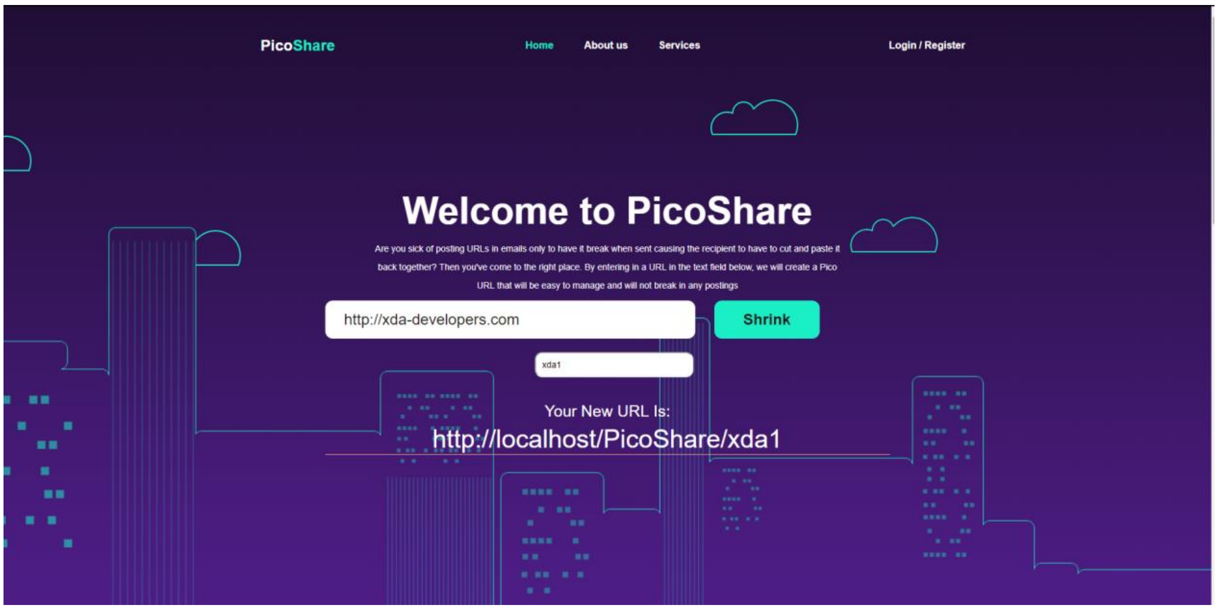
Welcome to PicoShare

Are you sick of posting URLs in emails only to have it break when sent causing the recipient to have to cut and paste it back together? Then you've come to the right place. By entering in a URL in the text field below, we will create a Pico URL that will be easy to manage and will not break in any postings

http://xda-developers.com

xda

Alias unavailable, Please try again



Welcome to PicoShare

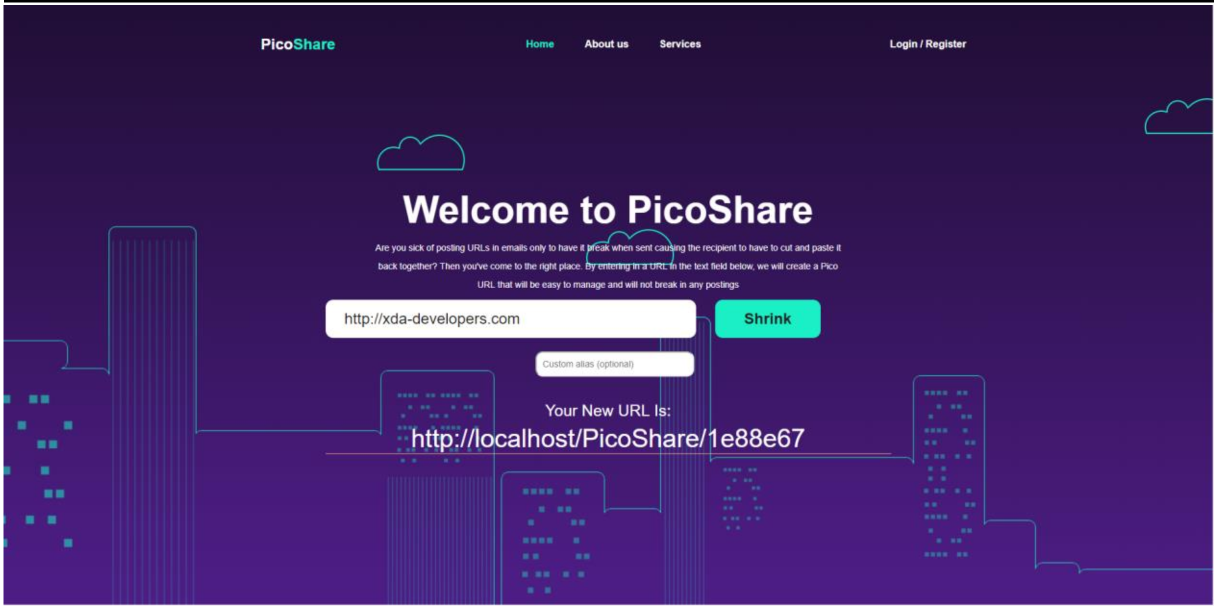
Are you sick of posting URLs in emails only to have it break when sent causing the recipient to have to cut and paste it back together? Then you've come to the right place. By entering in a URL in the text field below, we will create a Pico URL that will be easy to manage and will not break in any postings

http://xda-developers.com

xda1

Your New URL Is:

http://localhost/PicoShare/xda1



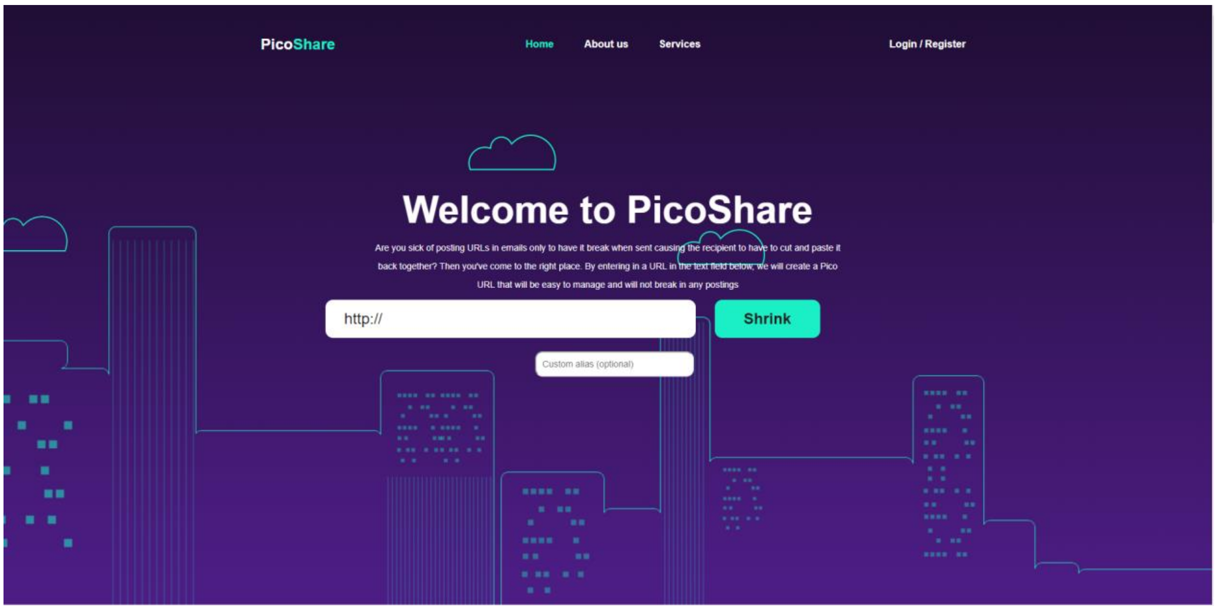
Welcome to PicoShare

Are you sick of posting URLs in emails only to have it break when sent causing the recipient to have to cut and paste it back together? Then you've come to the right place. By entering in a URL in the text field below, we will create a Pico URL that will be easy to manage and will not break in any postings

http://xda-developers.com

Your New URL Is:

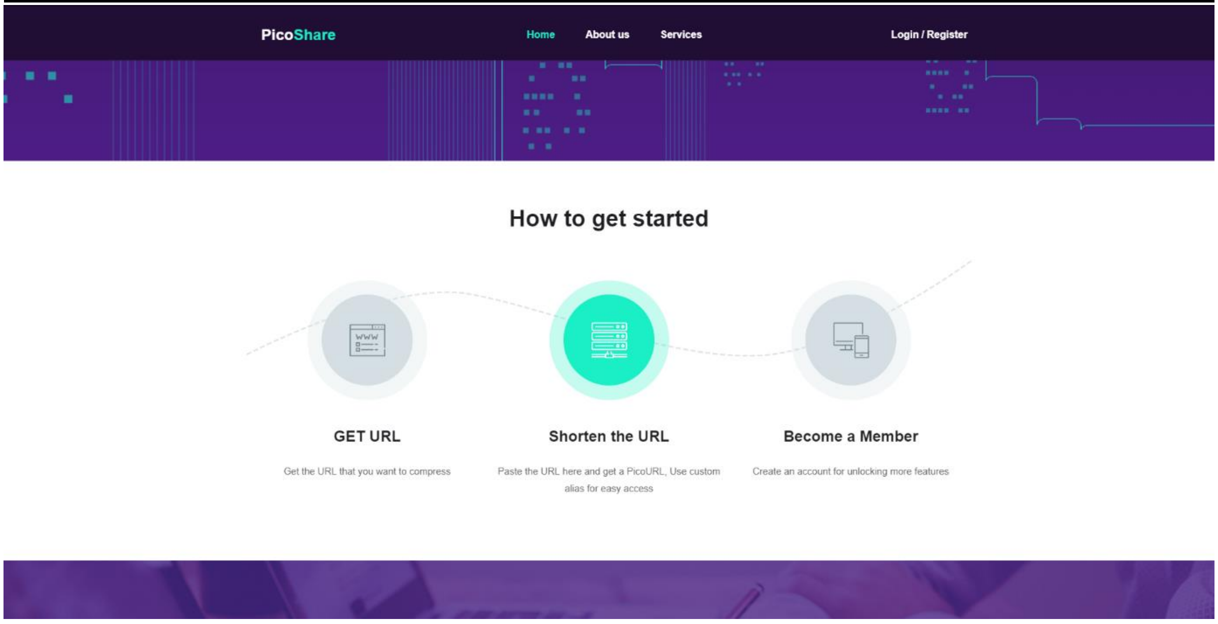
http://localhost/PicoShare/1e88e67



Welcome to PicoShare

Are you sick of posting URLs in emails only to have it break when sent causing the recipient to have to cut and paste it back together? Then you've come to the right place. By entering in a URL in the text field below, we will create a Pico URL that will be easy to manage and will not break in any postings

http://



How to get started

GET URL

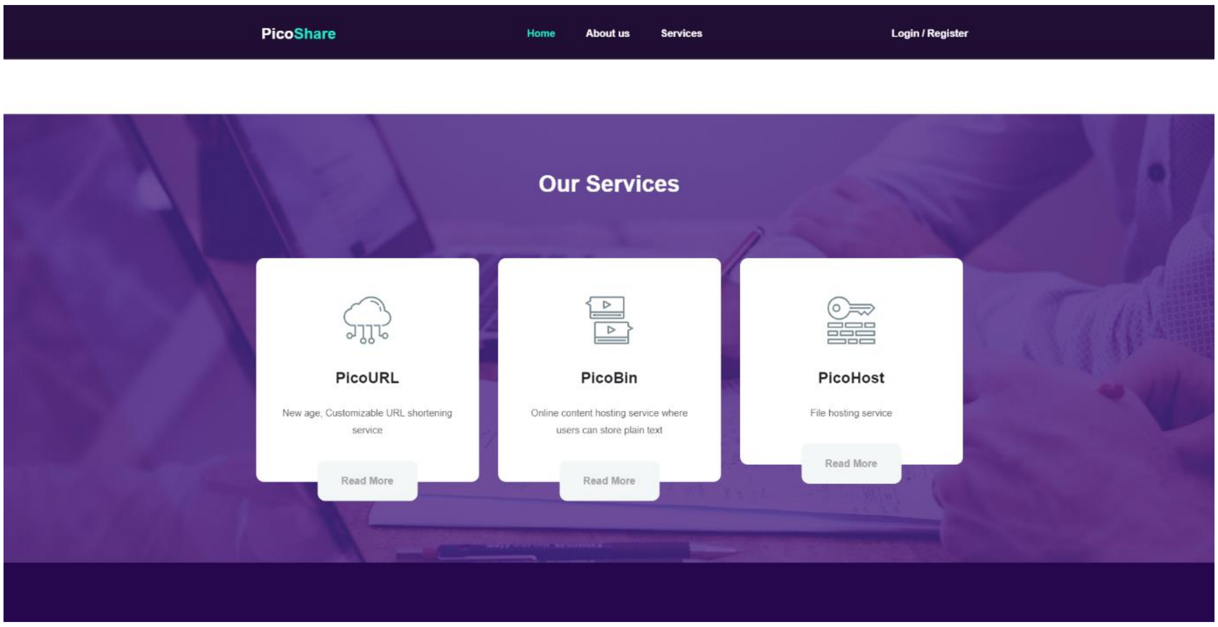
Get the URL that you want to compress

Shorten the URL

Paste the URL here and get a PicoURL, Use custom alias for easy access

Become a Member

Create an account for unlocking more features



Our Services

PicoURL

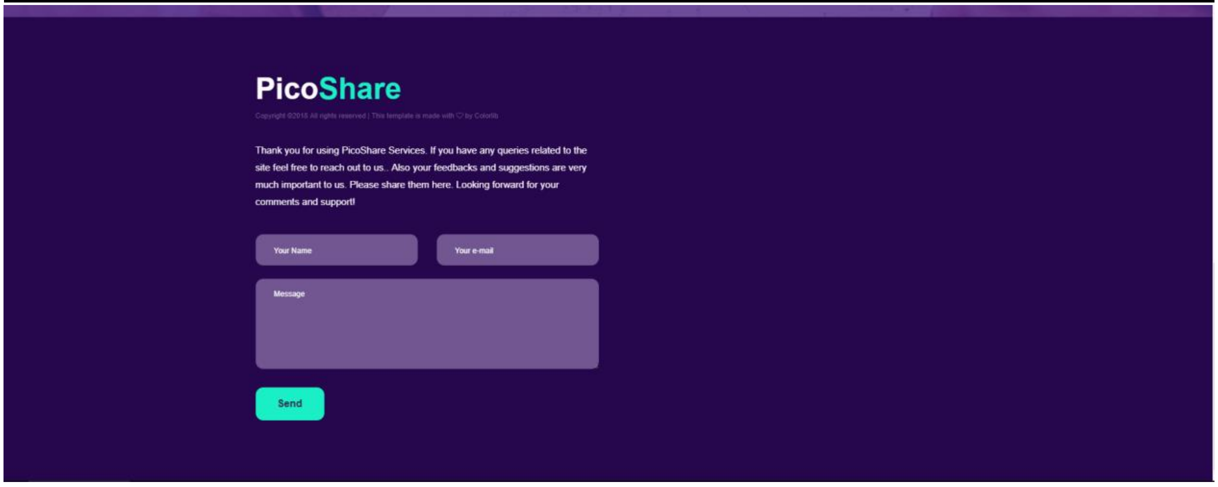
New age, Customizable URL shortening service

PicoBin

Online content hosting service where users can store plain text

PicoHost

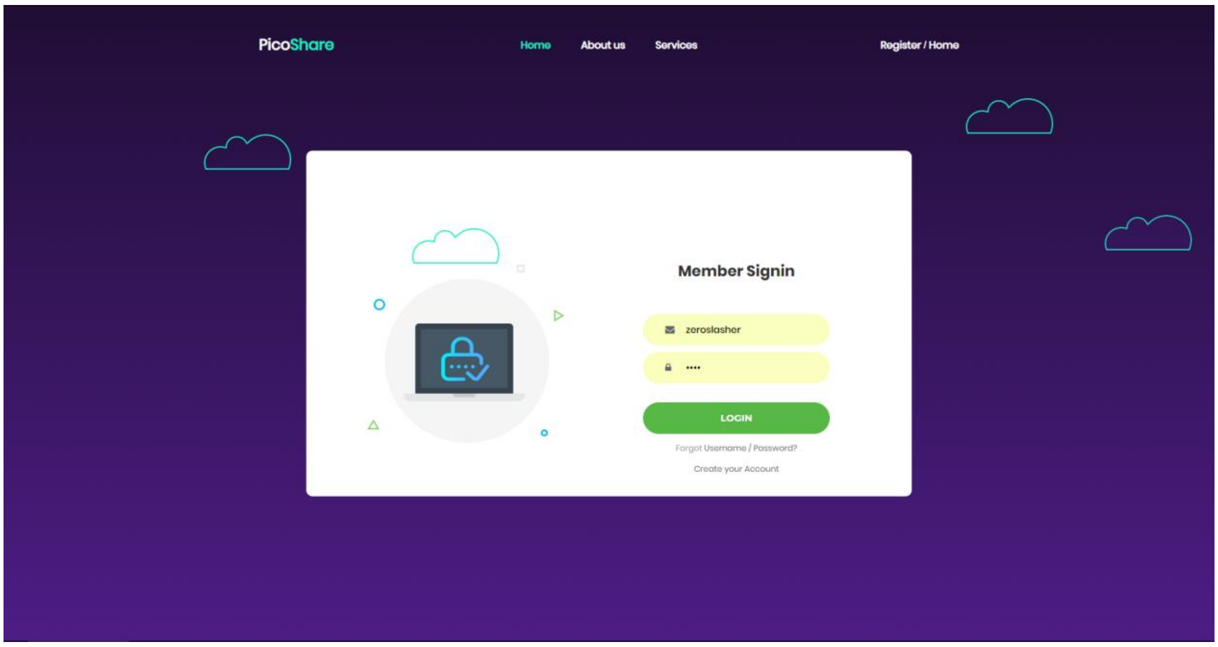
File hosting service

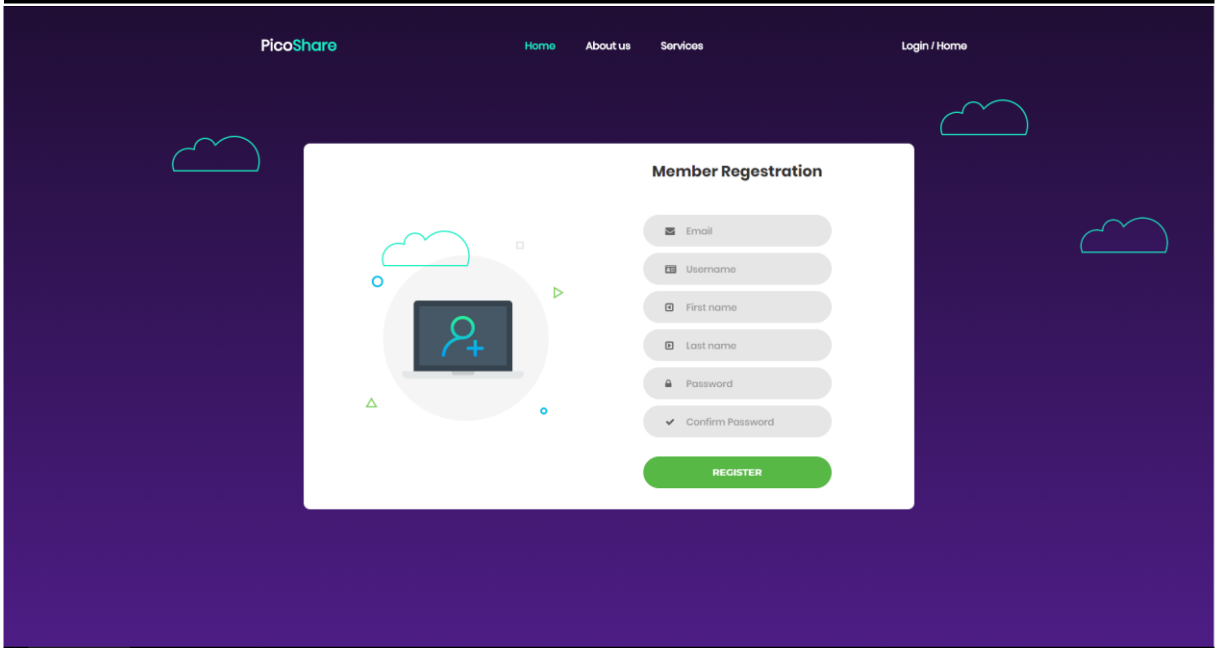


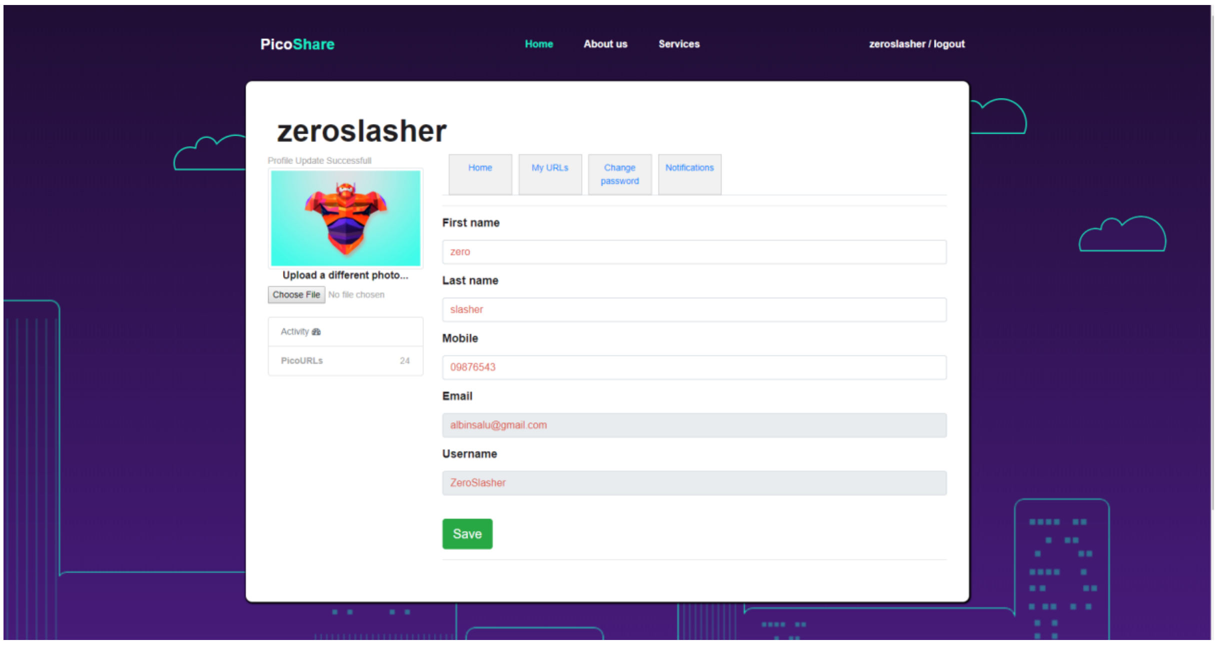
PicoShare

Copyright ©2018 All rights reserved | This template is made with ♡ by Colorlib

Thank you for using PicoShare Services. If you have any queries related to the site feel free to reach out to us.. Also your feedbacks and suggestions are very much important to us. Please share them here. Looking forward for your comments and support!







First name

zero

Last name

slasher

Mobile

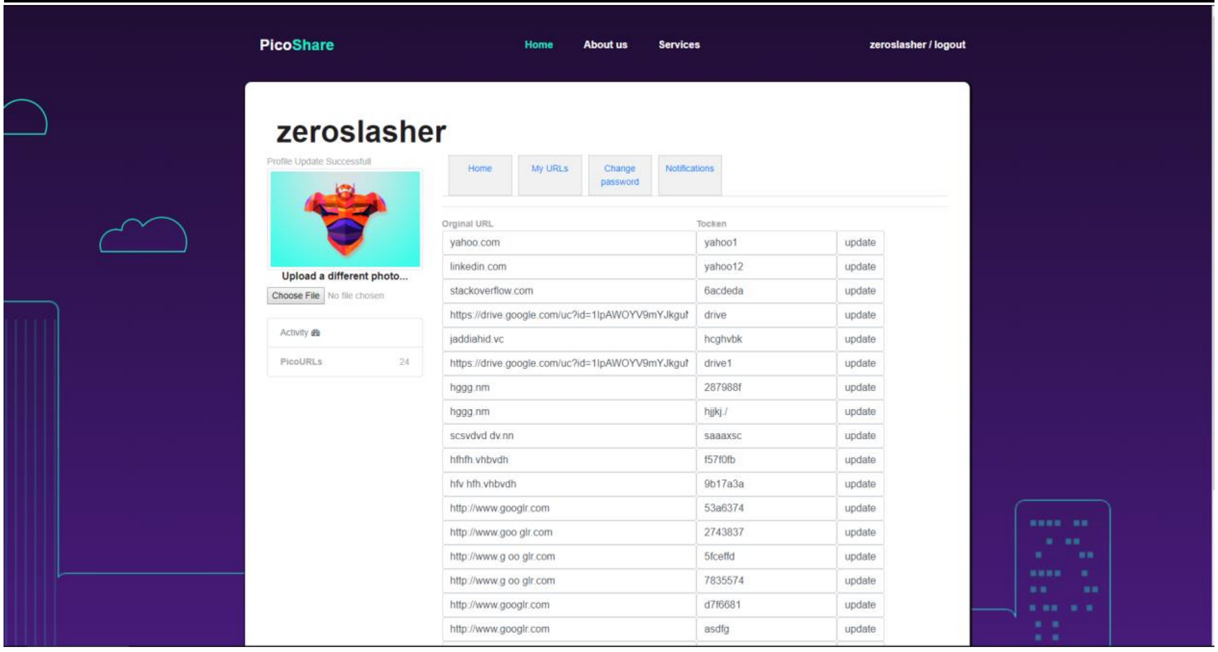
09876543

Email

albinsalu@gmail.com

Username

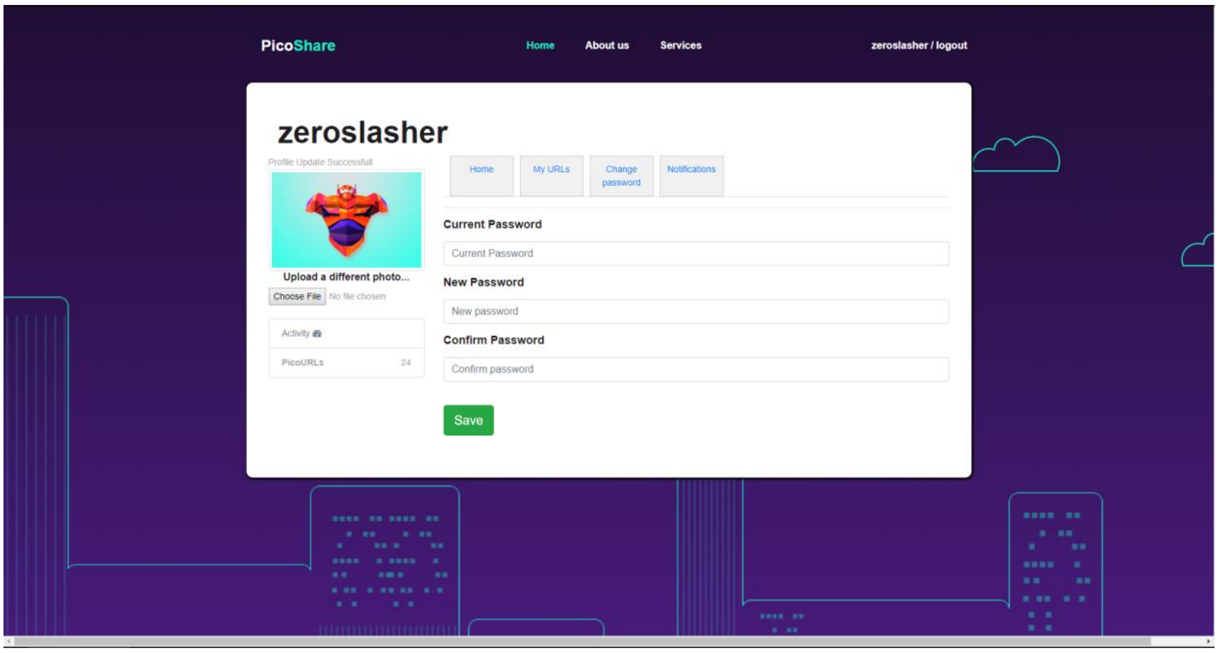
ZeroSlasher



zeroslasher

Profile Update Successfull

Orginal URL	Tocken	
yahoo.com	yahoo1	update
linkedin.com	yahoo12	update
stackoverflow.com	6acdeda	update
https://drive.google.com/uc?id=1lpAWOYV9mYJkgul	drive	update
jaddiahid.vc	hcghvbk	update
https://drive.google.com/uc?id=1lpAWOYV9mYJkgul	drive1	update
hggg.nm	287988f	update
hggg.nm	hjjkj./	update
scsvdvd dv.nn	saaaxsc	update
hfhfh.vhbvdh	f57f0fb	update
hfv hfh.vhbvdh	9b17a3a	update
http://www.googlr.com	53a6374	update
http://www.goo glr.com	2743837	update
http://www.g oo glr.com	5fceffd	update
http://www.g oo glr.com	7835574	update
http://www.googlr.com	d7f6681	update
http://www.googlr.com	asdfg	update



zeroslasher

Profile Update Successfull

Current Password

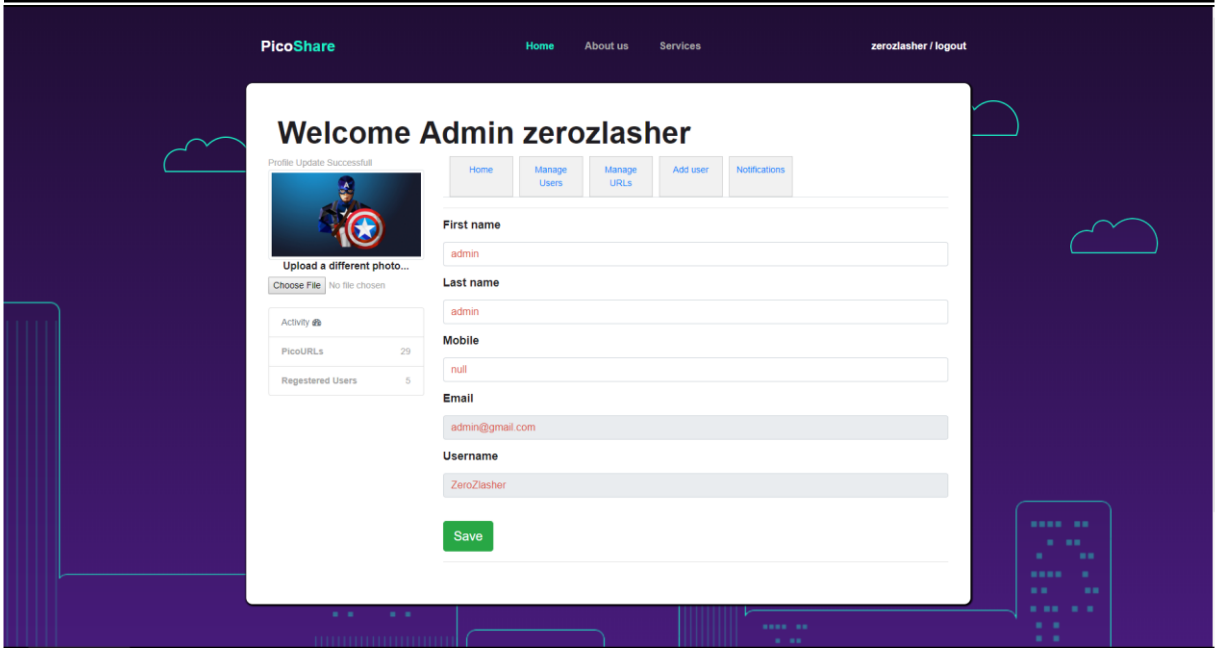
Current Password

New Password

New password

Confirm Password

Confirm password



Welcome Admin zerozlasher

Profile Update Successfull

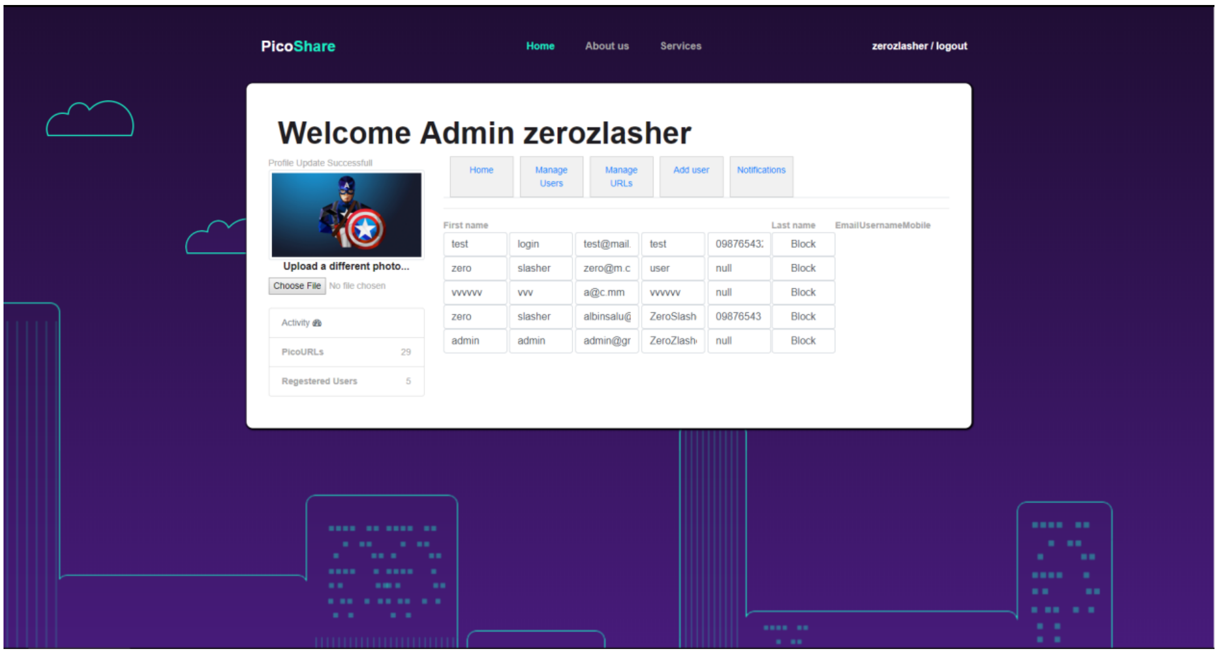
First name: admin

Last name: admin

Mobile: null

Email: admin@gmail.com

Username: ZeroZlasher



Welcome Admin zerozlasher

Profile Update Successfull

First name					Last name	EmailUsernameMobile
test	login	test@mail.	test	09876543:	Block	
zero	slasher	zero@m.c	user	null	Block	
vvvvvv	vvv	a@c.mm	vvvvvv	null	Block	
zero	slasher	albinsalu@	ZeroSlash	09876543	Block	
admin	admin	admin@gr	ZeroZlash	null	Block	