**CHAPTER 1**

**COMPANY PROFILE**

TechCiti has become a pioneer in providing distinguished end-to-end IT infrastructure solutions to its customers through our business functions maximizing customer engagement with personalized services. We believe that today more than ever, businesses are dependent on technology solutions.

**1.1 CSR Program**

We recognize the importance of implementing the right solution for your business. We offer a wide range of services to build a solution that is right for your business needs.

Every business, no matter the size, needs advice and support. We have several years of technical experience and have accumulated a wealth of IT infrastructure knowledge. Our free consultation service helps you to establish your requirements. We will be with you every step of the way, from product selection through to configuration and installation.

To stay relevant to our customers, it is important that we continuously demonstrate our ‘Value Creating Ability’ by:-

1." Helping our customers choose the right technology "

2. " Bring competitive commercials to the discussion table "

3." Remaining predictable in our execution "

4. " Backing it with excellent post-sales services "

TechCiti has become a pioneer in providing distinguished end-to-end IT infrastructure solutions to its customers through our business functions maximizing customer engagement with personalized services. We believe that today more than ever, businesses are dependent on technology solutions.

We have successfully established in our business functions across 12 + major cities across PAN India with over 56 + satisfied corporate customers.

**Vision** Our vision is to enable people and organizations realize their potential reinventing their engagement in defining the future using - technology.

**Mission** Our mission is to achieve the leading position as a distinguished & absolute end-to-end information technology infrastructure & service provider. We want to develop with profitable growth through superior Customer service, Innovation, Quality and Commitment.

**CHAPTER 2**

**DEPARTMENT**

Department of Information Science and Engineering functioning since 2001 with an intake of 120 and offering BE degree with a vision to mould the students with basic engineering knowledge, necessary IT skills to provide young dynamic engineers with high caliber and commitment towards the development of information and knowledge based solutions for the real world paradigms

The department is one of the oldest of the college, which is affiliated to VTU, Belagavi, approved by AICTE and was accredited by NBA. The department has highly professional and enthusiastic team of academicians interacting with current research activities. Faculties are always dedicated and devoted towards the comprehensive development of their students by training them psychologically through technical competencies globally. The department constantly encourages, monitors and counsels the students to develop personal qualities and competences.

The department has well equipped infrastructure support with state of art teaching aids for quality based education. The department is committed to encourage students and faculties to carry out innovative research in the field of IT and CSE, keeping excellence in focus and deliver quality service to match the needs of the technical education, industry and society. The department provides fundamental knowledge, technical skill, humanity and ethical practices to identify complex engineering problems by designing and developing solutions with modern tools.

The department provides opportunities to students to exhibit the talents, leadership qualities through departmental forum and CSI student chapters. The department was awarded as the Best Student chapter during the year 2015 for students’ involvement and commitment. The department organizes national conferences, Seminars, Student Symposium, youth festivals, short time training programs and value added courses. This provides wide range of opportunities for students and faculties to bring out their potential and innovative skills in various fields.

The department provides opportunities to excel themselves in Internship program. To bridge the gap between academics and industry needs, the department has interaction and MOU with leading technology domain training and development industries like EMC2 academic alliance

Internship Report and Infosys campus connect programs. The department prepares the students to pursue leadership, technical and management positions in various companies. This gets a successful placement at leading companies. The department has excellent placement records with reputed MNCs. Our Alumnus comprise of entrepreneurs, pursing higher education, involved in research activities in India and abroad.

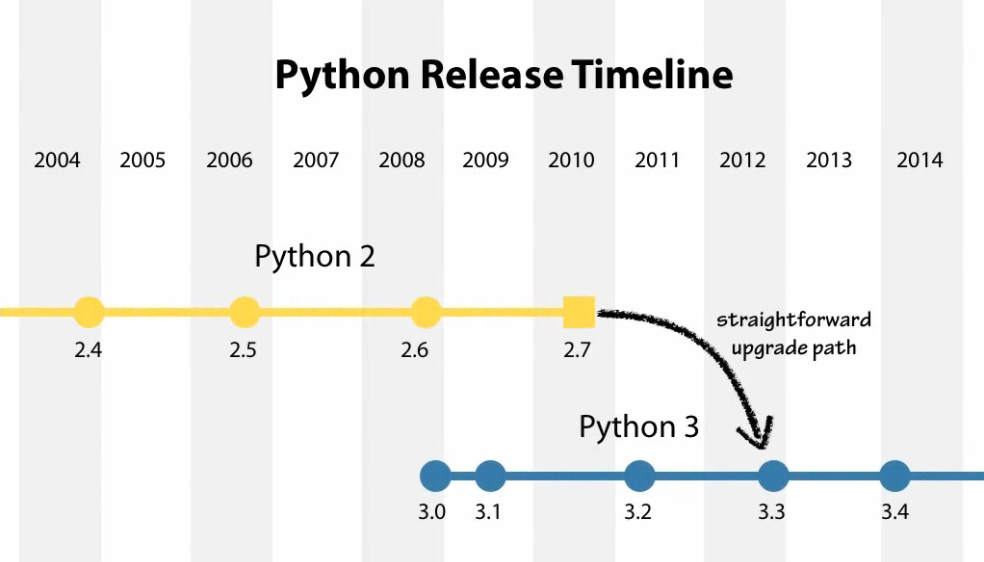
**CHAPTER 3**

**TASK PERFORMED**

**3.1 Introduction to PYTHON**

* Python is a high-level, interpreted, interactive and object-oriented, scripting language.
* This general purpose programming language is developed and designed by Guido van Rossum
* Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
* Python is derived from many other languages, including ABC, Modula-3, C, C++, ALGOL-68, SmallTalk, and Unix shell and other scripting languages.
* Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.
* Python is designed to be highly readable which uses English keywords frequently where as other languages use punctuation and it has fewer syntactical constructions than other languages.
* Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code
* It features a dynamic type system and strong type system(no implicit data type conversion)
* It has automatic memory management .
* It is Byte Code Interpreted Language.

**3.2 Different Versions of python**



**3.3 Features of Python**

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution. Python's design offers some support for functional programming in the Lisp tradition. It has filter, map, and reduce functions; list comprehensions, dictionaries, sets and generator expressions.

The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

**3.4 Applications of Python**

* web development (server-side),
* software development,
* mathematics,
* system scripting

What can Python do?

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-orientated way or a functional way.

**3.5 Django:**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

With Django, you can take Web applications from concept to launch in a matter of hours. Django takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Ridiculously fast: Django was designed to help developers take applications from concept to completion as quickly as possible.

Fully loaded: Django includes dozens of extras you can use to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks — right out of the box.

Reassuringly secure: Django takes security seriously and helps developers avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery and click jacking. Its user authentication system provides a secure way to manage user accounts and passwords.

Exceedingly scalable: Some of the busiest sites on the planet use Django’s ability to quickly and flexibly scale to meet the heaviest traffic demands.

The Django project's stability, performance and community have grown tremendously over the past decade since the framework's creation. Detailed tutorials and good practices are readily available on the web and in books. The framework continues to add significant new functionality such as [database migrations](https://docs.djangoproject.com/en/dev/topics/migrations/) with each release.

I highly recommend the Django framework as a starting place for new Python web developers because the official documentation and tutorials are some of the best anywhere in software development. Many cities also have Django-specific groups such as [Django District](http://www.meetup.com/django-district/), [Django Boston](http://www.meetup.com/djangoboston/) and [San Francisco Django](http://www.meetup.com/The-San-Francisco-Django-Meetup-Group/) so new developers can get help when they are stuck.

There's some debate on whether [learning Python by using Django is a bad idea](http://www.jeffknupp.com/blog/2012/12/11/learning-python-via-django-considered-harmful/). However, that criticism is invalid if you take the time to learn the Python syntax and language semantics first before diving into web development.

**3.6 SQLite:**

Sqlite is a c library that provides a lightweight disk-based database that doesn’t require a separate server process and allows accessing the database using a nonstandard variant of the sql query language. Some applications can use sqlite for internal data storage. It’s also possible to prototype an application using sqlite and then port the code to a larger database such as postgresql or oracle.

Usually your sql operations will need to use values from python variables. You shouldn’t assemble your query using python’s string operations because doing so is insecure; it makes your program vulnerable to an sql injection attack. Instead, use the db-api’s parameter substitution. Put ? As a placeholder wherever you want to use a value, and then provide a tuple of values as the second argument to the cursor’s [execute()](https://docs.python.org/3/library/sqlite3.html#sqlite3.Cursor.execute) method.

Sqlite3.version: the version number of this module, as a string. This is not the version of the sqlite library.

Sqlite3.version\_info: the version number of this module, as a tuple of integers. This is not the version of the sqlite library.

Sqlite3.sqlite\_version: the version number of the run-time sqlite library, as a string.

Sqlite3.sqlite\_version\_info: the version number of the run-time sqlite library, as a tuple of integers.

Sqlite3.parse\_decltypes:this constant is meant to be used with the detect\_types parameter of the [connect()](https://docs.python.org/3/library/sqlite3.html#sqlite3.connect) function.

Setting it makes the [sqlite3](https://docs.python.org/3/library/sqlite3.html#module-sqlite3) module parse the declared type for each column it returns. It will parse out the first word of the declared type, i. E. For “integer primary key”, it will parse out “integer”, or for “number(10)” it will parse out “number”. Then for that column, it will look into the converters dictionary and use the converter function registered for that type there.

Sqlite3.parse\_colnames: this constant is meant to be used with the detect\_types parameter of the [connect()](https://docs.python.org/3/library/sqlite3.html#sqlite3.connect) function.

Setting this makes the sqlite interface parse the column name for each column it returns. It will look for a string formed [mytype] in there, and then decide that ‘mytype’ is the type of the column. It will try to find an entry of ‘mytype’ in the converters dictionary and then use the converter function found there to return the value. The column name found in [cursor.description](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3.Cursor.description" \o "sqlite3.Cursor.description) is only the first word of the column name, i. E. If you use something like 'as "x [datetime]"' in your sql, then we will parse out everything until the first blank for the column name: the column name would simply be “x”.

Sqlite3.connect(database[, timeout, detect\_types, isolaion\_level, check\_same\_thread, factory, cached\_statements, uri])

Opens a connection to the sqlite database file database. By default returns a [connection](https://docs.python.org/3/library/sqlite3.html#sqlite3.Connection) object, unless a custom factory is given.

Database is a [path-like object](https://docs.python.org/3/glossary.html#term-path-like-object) giving the pathname (absolute or relative to the current working directory) of the database file to be opened. You can use ":memory:" to open a database connection to a database that resides in ram instead of on disk.

When a database is accessed by multiple connections, and one of the processes modifies the database, the sqlite database is locked until that transaction is committed. The timeout parameter specifies how long the connection should wait for the lock to go away until raising an exception. The default for the timeout parameter is 5.0 (five seconds).

For the isolation\_level parameter, please see the [isolation\_level](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3.Connection.isolation_level" \o "sqlite3.Connection.isolation_level) property of [connection](https://docs.python.org/3/library/sqlite3.html#sqlite3.Connection) objects.

Sqlite natively supports only the types text, integer, real, blob and null. If you want to use other types you must add support for them yourself. The detect\_types parameter and the using custom converters registered with the module-level [register\_converter()](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3.register_converter" \o "sqlite3.register_converter) function allow you to easily do that.

Detect types defaults to 0 (i. E. Off, no type detection), you can set it to any combination of [parse\_decltypes](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3.PARSE_DECLTYPES" \o "sqlite3.PARSE_DECLTYPES) and [parse\_colnames](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3.PARSE_COLNAMES" \o "sqlite3.PARSE_COLNAMES) to turn type detection on.

By default, check\_same\_thread is [true](https://docs.python.org/3/library/constants.html#True) and only the creating thread may use the connection. If set [false](https://docs.python.org/3/library/constants.html#False), the returned connection may be shared across multiple threads. When using multiple threads with the same connection writing operations should be serialized by the user to avoid data corruption.

By default, the [sqlite3](https://docs.python.org/3/library/sqlite3.html#module-sqlite3) module uses its [connection](https://docs.python.org/3/library/sqlite3.html#sqlite3.Connection) class for the connect call. You can, however, subclass the [connection](https://docs.python.org/3/library/sqlite3.html#sqlite3.Connection) class and make [connect ()](https://docs.python.org/3/library/sqlite3.html#sqlite3.connect) use your class instead by providing your class for the factory parameter.

Consult the section [sqlite and python types](https://docs.python.org/3/library/sqlite3.html" \l "sqlite3-types) of this manual for details.

The [sqlite3](https://docs.python.org/3/library/sqlite3.html#module-sqlite3) module internally uses a statement cache to avoid sql parsing overhead. If you want to explicitly set the number of statements that are cached for the connection, you can set the cached statements parameter. The currently implemented default is to cache 100 statements.

If uri is true, database is interpreted as a uri. This allows you to specify options. For example, to open a database in read-only mode you can use:

Db=sqlite3.connection (‘file:path/to/database?mode=ro’, uri=true)

**3.7 Cascading style sheets (css)**

CSS is an abbreviation for Cascading Style Sheets. CSS works with HTML and other

Markup Languages (such as XHTML and XML) to control the way the content is presented.

Cascading Style Sheets is a means to separate the appearance of a webpage from the content of a

webpage. CSS is a recommendation of the World Wide Web Consortium (the W3C). The W3C is

a consortium of web stakeholders: universities, companies such as Microsoft, Netscape and

Macromedia, and experts in many web related fields. The presentation is specified by styles, which

are presented in a style sheet. If you're familiar with word processing programs like Microsoft

Word, you have probably played around at least a little bit with styles. For example, when you

want to make the headline text of your document big and bold, the hard way to do it would be to

select the text, select a font face and weight, and select the color. The easier way to do it (presuming

your document has more than one headline) is to create a "rule", or style, for all the headlines in

your document. Then all you have to do is to make one rule, and keep on applying that to all your

headers. CSS in its most basic form works exactly like this. Instead of using <FONT> tags over

and over again to control little sections of your page, you can establish some rules to apply

globally, to a single page or all the pages on your site. CSS is a great time saver.

The cascade part of CSS means that more than one style sheet can be attached to a

document, and all of them can influence the presentation. For example, a designer can have a

global style sheet for the whole site, but a local one for say, controlling the link color and

background of a specific page. Or, a user can use her own style sheet if she has problems seeing

the page, or if she just prefers a certain look. Cascading Style Sheets, fondly referred to as CSS, is

a simple design language intended to simplify the process of making web pages presentable. CSS

handles the look and feel part of a web page. Using CSS, you can control the color of the text, the

style of fonts, the spacing between paragraphs, how columns are sized and laid out, what

background images or colors are used, as well as a variety of other effects. CSS is easy to learn

and understand but it provides a powerful control over the presentation of an HTML document.

3.7.1 Who creates and maintains CSS?

CSS is created and maintained through a group of people within the W3C called the CSS Working

Group. The CSS Working Group creates documents called specifications. When a specification

has been discussed and officially ratified by the W3C members, it becomes a recommendation.

These ratified specifications are called recommendations because the W3C has no control over the

actual implementation of the language. Independent companies and organizations create that

software.

3.7.2 CSS Versions

Cascading Style Sheets level 1 (CSS1) came out of W3C as a recommendation in December 1996.

This version describes the CSS language as well as a simple visual formatting model for all the

HTML tags. CSS2 became a W3C recommendation in May 1998 and builds on CSS1. This version

adds support for media-specific style sheets e.g. printers and aural devices, downloadable fonts,

element positioning and tables.

3.7.3 CSS Syntax

A CSS comprises of style rules that are interpreted by the browser and then applied to the

corresponding elements in your document. A style rule is made of three parts:

• Selector: A selector is an HTML tag at which a style will be applied. This could be any tag

like <h1> or <table> etc.

• Property: A property is a type of attribute of HTML tag. Put simply, all the HTML

• attributes are converted into CSS properties. They They could be color, border, etc.

Value: Values are assigned to properties. For example, color property can have the value

either red or #F1F1F1 etc.

You can put CSS Style Rule Syntax as follows:

selector { property: value; }

Example: You can define a table border as follows

table{ border :1px solid #C00; }

**CHAPTER 4**

**REFLECTION**

Fitness center management application to maintain and manage the gym customers information or data is developed using python in Django framework .I have used python as server side language and sqlite is used as back end design . I have used cascading style sheets as front end tool. this application helps the gym management to access , retrieve and maintain their customers information in a easy way there by reducing the burden of recording all the information in books or record physically.

APPLICATION DEVELOPMENT

**4.1 Project Execution:**

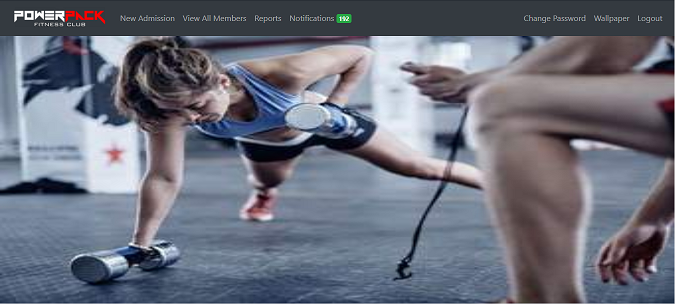
4.1.1 Home module

fig 4.1.1 shows the home page of gym management

4.1.2 Admin login module:

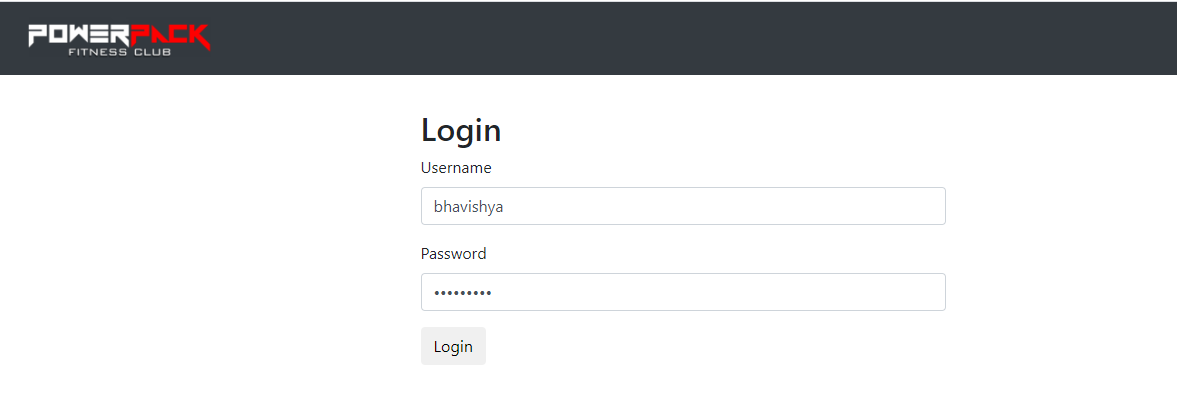


fig 4.1.2 shows the credentials required for the admin to log in

4.1.3: Dues:

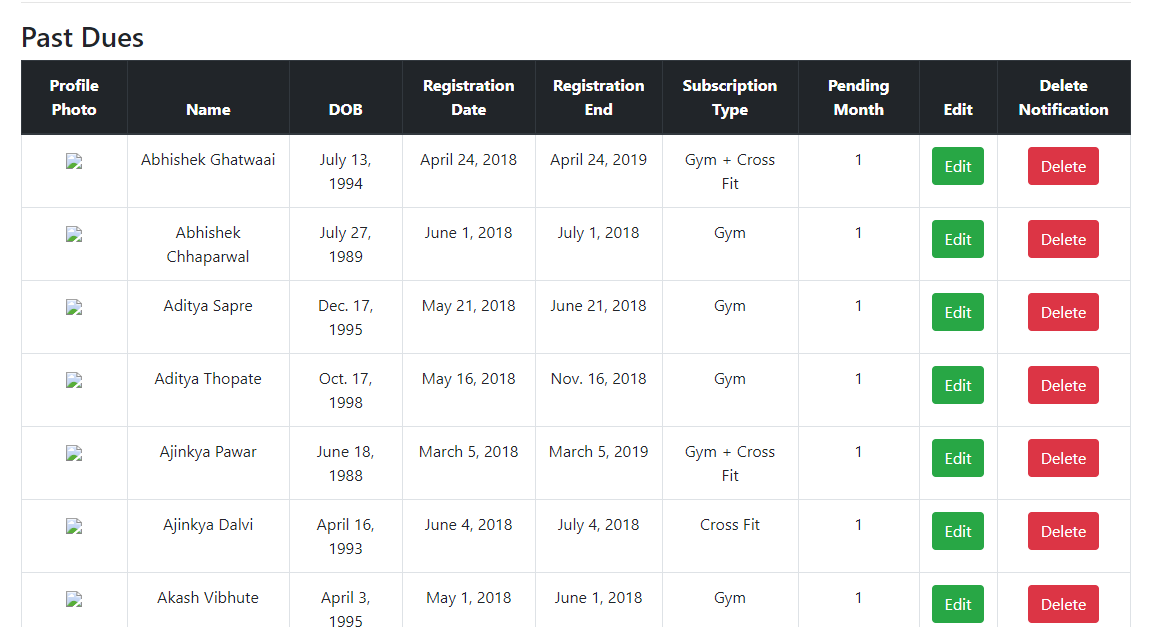


fig 4.1.3 shows the details about the members who has their fee dues if the customer is no longer a member in the gym then the admin can delete his information by clicking the delete button corresponding to his/her information

4.1.4: view of database

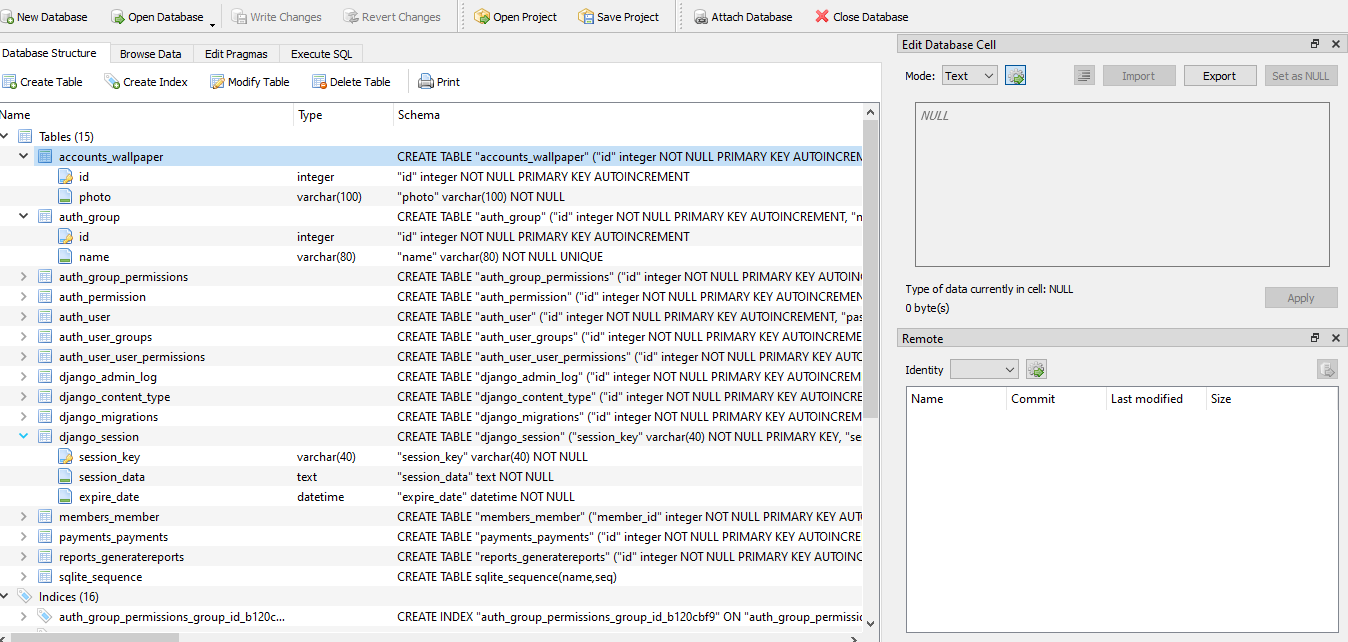
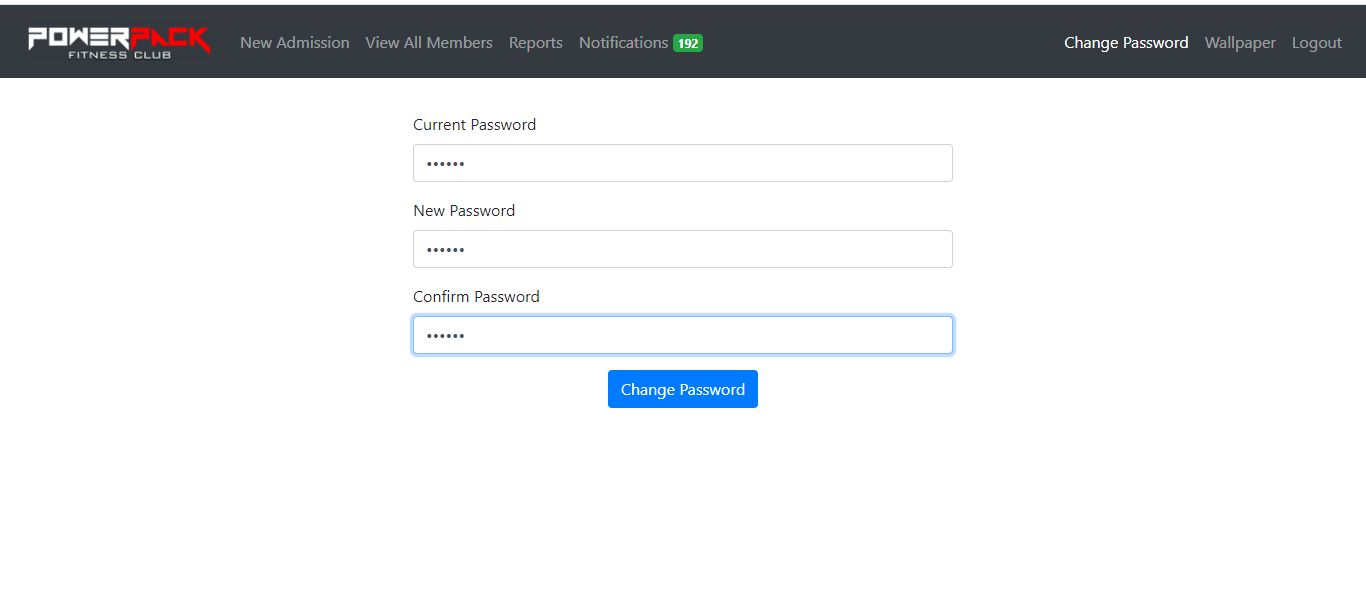


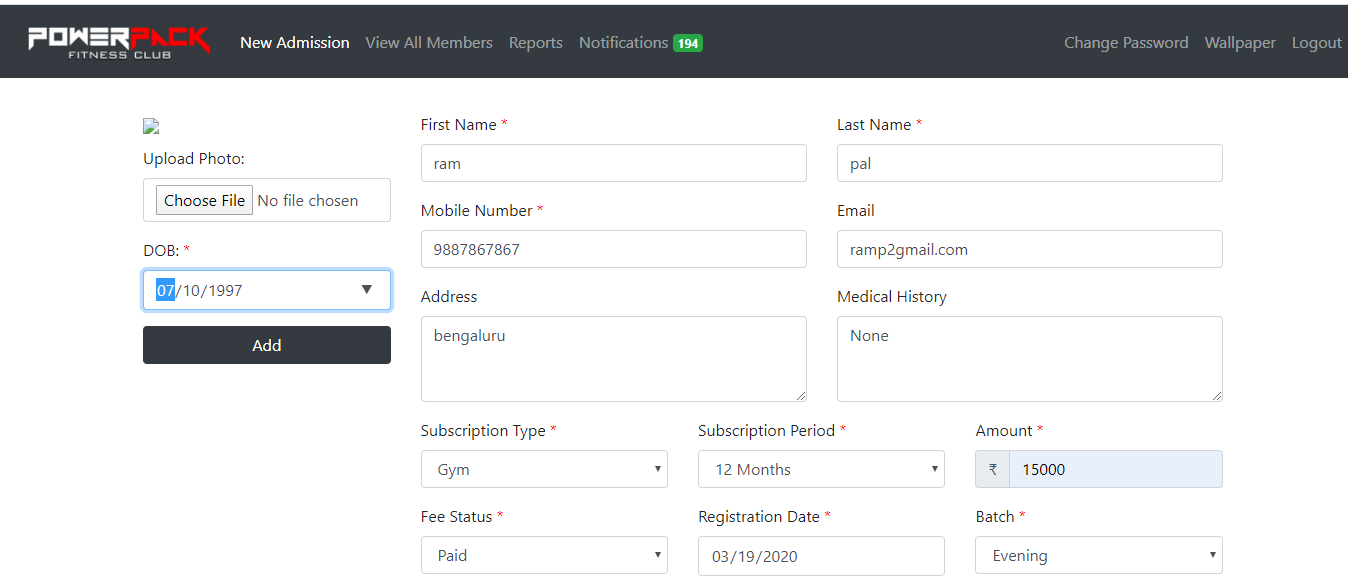
fig 4.1.4 shows the details about the tables

4.1.5 Change password



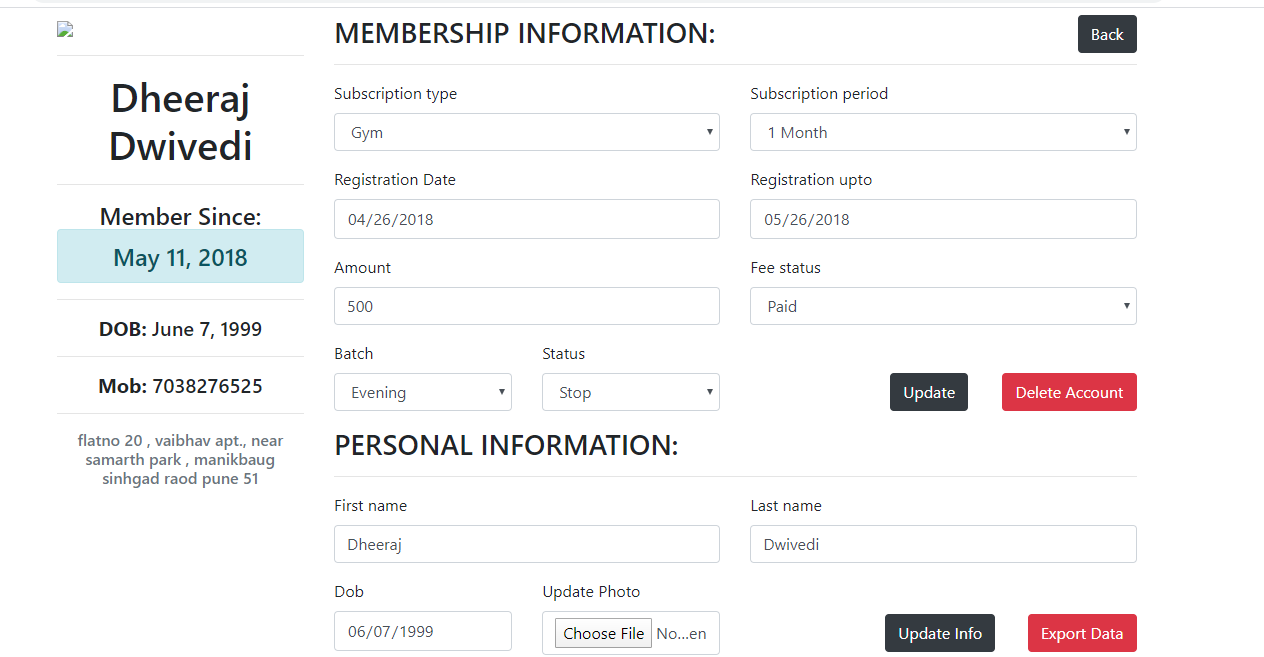
Change password module which is shown in above fig 4.1.5 helps the admin to keep changing his/her password for the various security reasons.

4.1.6: Add member



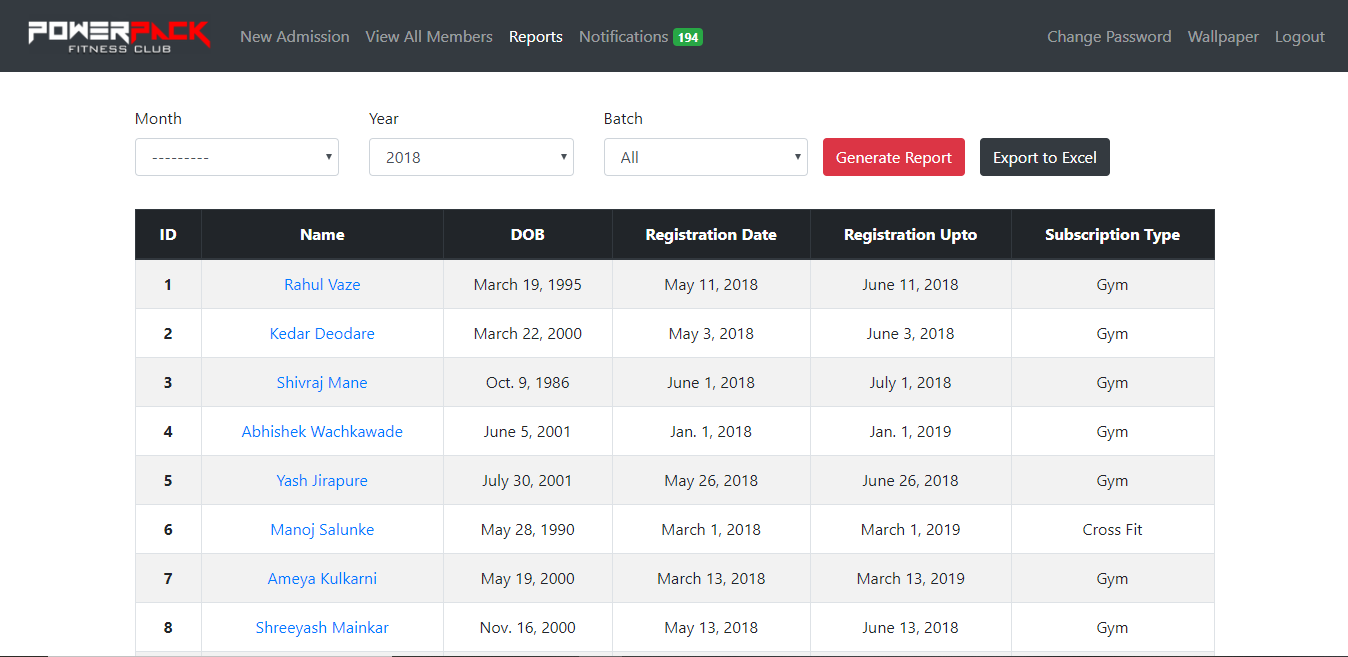
Above fig 4.1.6 shows the essential information need to be provided by the joining customer in order to be a gym member where he/she need to specify the subscription type, period and batch.

4.1.7: Member information



when the customer leaves the gym after finishing subscription period the related data of the data should be deleted, this can be done by the admin as shown in fig 4.1.7

4.1.8: Report



If the admin wants to know the statistics of how many members registered we can generate a report based on particular year and based on particular month which gives the precise view of membership.

**4.2 Database**

4.2.1: admin table

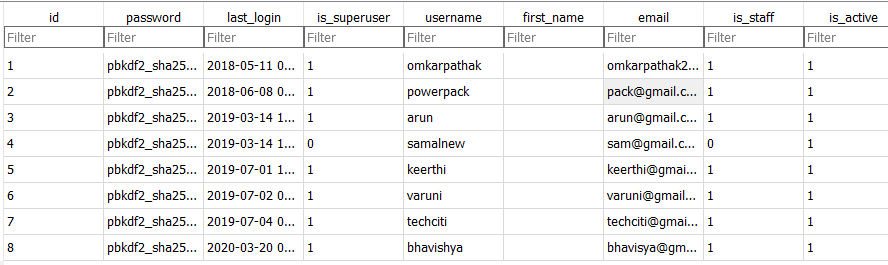


fig 4.2.1 shows the admin table and their details

4.2.2: payments table:

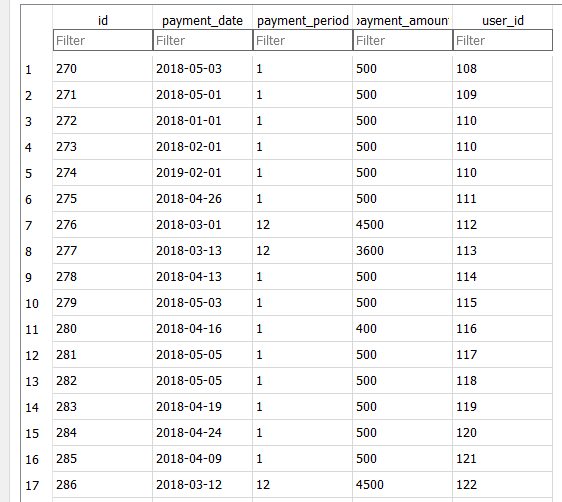


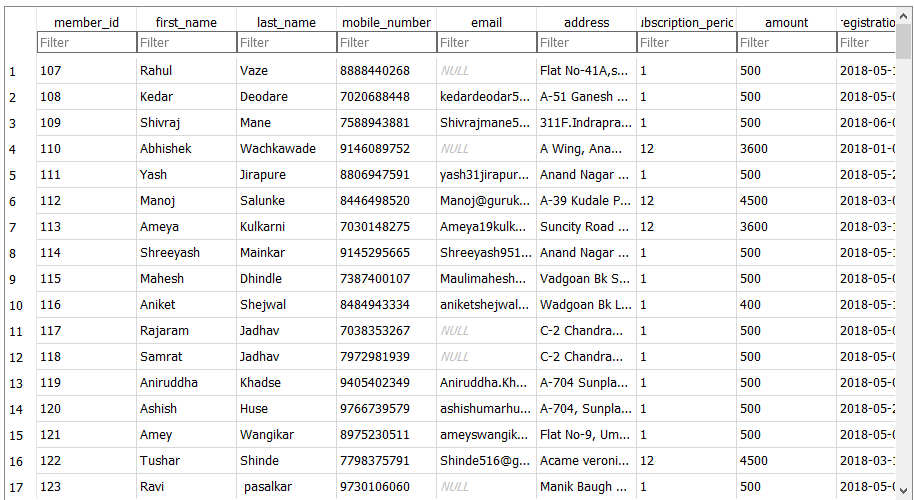
fig 4.2.2 shows the table that contains the details about the payments

4.2.3: permissions table:



The above table 4.2.3 shows the permissions of admin

4.2.4: Members table:



Above table 4.2.4 shows the users who are members of fitness center management