VAIBHAV SHAH

C-501, Parijat Society, Opp. Kirloskar Oil Engines,

Bopodi, Pune -411003.

Mobile: +91-9975846199.

Landline: 020-25824460.

E-mail: vaibhavshah846@gmail.com.

Career Objectives:

1. To work in the Software industry and gain an in depth knowledge in various domains.

2. To work towards making the Information technology domain more secure and reliable.

3. To gain cross-product knowledge through providing services.

4. To empower nation's economy and infrastructure through use of technology.

Educational Details:

Qualification	Stream	Institute	Year	Results
Bachelor of Engineering (B.E.).	Computer Science	MIT College Of Engineering, Pune.	2013	69.80% First Class with distinction.
Junior College.	Science	Fergusson College, Pune-411004.	2009	80% overall. 82% in Physics, Chemistry, Mathematics
School.		St. Joseph's Boys High School, Pune	2007	84.46%

Work Experience:

Company	Position	Products	Number of years	Technologies
ACI Worldwide, Inc.	Associate Software Engineer	Postilion and Universal Payments Platform	1year 9 months	Core Java, Groovy Scripting, Postilion Framework, UPF Framework, Advanced Java Basics.
ACI Worldwide, Inc.	Consultant Contractor	Universal Payments Platform	2.5 months	Universal Payments
Ebex Consulting (Present)	Python Backend Developer	Convincely	3 months and counting	Python, Django, DjangoREST Framework, AngularJS, JavaScript, AJAX, JSON, jQuery

While at ACI, I worked extensively on **Postilion migration projects and UPF Adapters Projects**.

I gained good understanding about **Postilion Framework, UPF Framework and test harness**.

At Ebex, I have been working in **Website Conversion Analytics Domain** and involved in the development of the product **Convincely**.

Skills and Programming Languages Known:

Advanced	C, Core Java, Groovy
Intermediate	PL/SQL, VB 6.0., **Python**, **Django**, **DjangoREST Framework**, **AngularJS**, **JavaScript**, **AJAX**, **JSON**, **jQuery**
Basics	HTML, PHP, CSS, XML, WSDL, XSD, etc.

Knowledge of version control tool **Perforce, GIT repository, Bit bucket, JIRA**, etc. and basic knowledge of test coverage and report generation tool **Clover**.

I am experienced with **Linux Operating System and its commands**.

I worked with tools such as **Microsoft SQL Server 2000 and 2010, Squirrel, Apache Tomcat Server, etc**.

Projects Undertaken:

TE Semester1 Project: Supermarket Analysis (Using VB and Oracle).

Description: We were supposed to completely automate a supermarket scenario. The system had to help the purchase manager to place orders and also make important inventory decisions by using the reports generated by the system. The database was designed and implemented in Oracle whereas the front-end user interface was designed in VB 6.0.

TE Semester2 Project: Snake Game in .NET.

Description: The aim was to design the Snake Game which also included several levels of speed of snake and placement of mazes.

ACI Worldwide Projects: Migrations and UPP (Total 2 years' experience)

Migrations: Worked with the migrations team for the first 7 months. S1's Postilion framework had a major upgrade from Postilion v4.3 to Real-time Framework v5.5. The clients were having the v4.3 of the framework and were running components built on v4.3.The task of the migrations team was to upgrade the client's components to make them work on Real-time Framework v5.5.This migration of the components was carried out by following a predefined process that included:

- Setting up and assessing the current client environment.
- Running the existing test suite to ensure all test cases run successfully.
- Migration of the test cases.
- Migration of the source code.
- Migration of the documentation.
- Migrating additional components.
- Performing clean install testing, upgrade testing, rollback testing, etc.
- Checking in the migrated components to Perforce after all testing is successful.
- QA and Release Management.
- Clover Reports Generation.

Successfully completed release of 4 migration projects:

- Euronet VisaBase12
- Wakefern SolutranReturnedCheckLoader
- Wakefern EmePosEntity
- Wakefern Oracle Financials RPC Extract

UPF: It stands for Universal Payments Framework which is a new product that ACI acquired as a result of acquisition of Distra. It mainly uses the Groovy Scripting language. I went through 2 months product training. The projects undertaken were as follows:

- Configuration of Endpoints for Singapore's Maybank.
- Configuration of Endpoints for Square One Bank adapters (Fundtech Host).
- Configuration of Endpoints for Dollar Bank adapters (Nautilus Host).
- Configuration of Endpoints for Banco De Bogota adapters (Vasco Host).

- Configuration of Endpoints for Dollar Bank Authentication Adapter (Vasco)
- Configuration of Endpoints for East West Bank (Metavante Host).
- Configuration of Endpoints for East West Bank (AFS FIS Host).
- Configuration of Endpoints for Al Rajhi Bank Adapters (Vasco IdentiKey Host).
- Configuring and testing the newly added SWIFT Block Field Functionality

Out of the above projects I have led the entire development of Dollar Bank Adapters, Banco De Bogota adapters, East West Bank Adapters and Al Rajhi Bank adapters as a team lead. The Dollar Bank Adapters and Al Rajhi Bank adapter's projects were completed ahead of schedule and the team work was appreciated by the management. None of the other projects slipped ahead of schedule.

While working on adapters, I have gained immense knowledge of various protocols and types of interfaces that banks use for their Host Systems which are provided by different vendors such as Fundtech, Nautilus, Vasco, Metavante, etc.

Most of the above clients had the requirement to set up either **1-way or 2-way SSL with Certificate Exchange between the ACI Systems and the Bank Host. This task was carried out primarily by me**. As UPF is mostly used with Linux systems, I also gained some good Linux skills.

Ebex Projects (Present 3 months and counting):

Convincely:

Convincely is a product of **Ebex Consulting**. The product allows customers to create Campaigns for their websites. They can set **Goals** for each of the webpages in their website (For Eg: How many customers click the Signup/Sign In button? How many people click the Social login with Facebook, Twitter? , How many people click Buy Now button? , etc.)

A **tracking script tracks the website for recording Unique Visitors** to that website and also the **number of Unique Visitors who accomplished the Goals set**. Based on this data Conversion Rates are measured on a daily basis and

reports are generated and displayed using a Graphical Representation for a particular date range.

The product also allows the customer to create different variations of his/her website and measure conversion counts for each Variation separately.

Tasks accomplished:

Optimization of the Identification Module:

The Identification module is responsible for scraping a website and identifying the CTA (Call to Action) elements out of it. The algorithm written was taking too much time to run. Thus, we needed to optimize it.

I performed a timing analysis on the algorithm by using the **Python line Profiler** to record the amount of time taken by different parts of the program to run.

The analysis revealed that the **maximum percentage of time was taken** by the **Regular Expression based string searches and the urlparse () method** call during the **creation of Anchor Objects.**

Replacement of the regular expression search with the python in operator and replacement of the urlparse () with simple String Breakup code led to a 10% improvement of the total time taken.

Resolved Issues with the Variations Page:

The Variations Page was having issues such as no actions happening on clicking the buttons such as (Play Variation, Pause Variation, Edit Variation, etc.)

Investigations revealed that the associated Django APIs (Play Variation API, Pause Variation API, Edit Variation API, etc.) were not returning the variationId for the variation on which these actions were being performed and thus, the angular ng-click was not working.

Modified these Django REST APIs to return the Variation Ids and Variation Status (active/inactive) along with the Variation name.

Implemented the Graph to display the Conversion Rates and Improvement Rates:

Developed the **Django Page Report REST API** that would calculate the Unique Visitor Count and the Conversion Count for each day in the input date range and return a consolidated day wise list of both unique_visitor and conversion_count for each day of the months for different variations.

This **data was then used to generate a Conversion rate and Improvement rate Line Graph** with different Lines representing different variations on the frontend.

Resolved Issues with the Tracking Script:

The tracking script is a script that gets emailed to a user when he/she creates a new campaign for his website. This script monitors the website to record unique visitors and conversion counts for each visitor of the website and stores these records in the backend which can then be used to generate Page Reports. The tracking script needed to be modified so that the following requirements are satisfied:

- Need to update jQuery $ with some unique identifier handler
- All Ajax calls must be asynchronous.
- JQuery must be loaded with no-conflict parameter.
- Js libraries must be loaded in the async mode.
- All APP & API urls should be set from the variable in 1 central location.
- All async calls should be fired once so there should be some kind of semaphore which should not block any page click operations.
- Need to check whether the jquery is loaded. If it's loaded check the version. If the version does not match the requirement then load the required version of the jQuery.

Implemented all the above features.

Page Report API not returning results for last month in the date range:

For an input date range spanning more than 2 months (Eg: March and April) and start_date>15 and end_date<15 (For Eg: 16th March 2016 to 14th April 2016), the Page Report API was not returning results for the last month (April in this case).

Investigations revealed that list of months to be processed were being populated with the help of a rule like: **rrule (MONTHLY, dtstart=strt_dt, until=end_dt)**.As the end_dt was less than 15, the last month was getting skipped from the list of months being processed. Rectified the issue by adding a boundary check for the last month in the month_list to match the month of end_dt.

Implement Google Sign In for the website:

Implemented the Google Sign In for Convincely using **OAUTH**. Built a **Django Social Login API** which would be called if the user has requested login via Google. This API would be provided with Email and Google **id_token** as input. The id_token would be verified using the **verify_token (id_token)** method of the **oauth2client Python library**. If the token verification is successful, then the user would be marked as logged in and a token would be returned for the session.

Updating Variation Traffic Distribution Percentage:

The traffic distribution percentage for Variations needed to be displayed for each Variation. Modified the **DjangoVariations API** to return the traffic distribution percentage for each Variation.

Page Report API was always returning 0 Unique Visitors:

Even though multiple unique visitors existed for a campaign, the **Django Page Report API** was returning zero for unique visitor count.

Investigations revealed that there was a problem with the logic of inserting the data for each day of a month in a Python dictionary. In place of the updating a day in a month with unique_visitor _count, the entire month was being updated. As a result the insertion of conversion_count for a particular day was completely replacing the unique_visitor_count.

Modified the **Django RESTAPI** to insert the data into a day of the month and not the entire month.

Wrong Use of Popping out Aims on the Aim selection Page:

The Aim selection worked with a JavaScript Array where each entry in the array was a key-value pair. The existing algorithm was written to pop-out the element whose key is passed to the pop (key) function. But in reality pop method only pops out the last element of the array, irrespective of the key passed. This **was a crucial bug that could have gone unidentified and would have caused a lot of failures in production environment**. I identified this issue while solving another issue and fixed it to avoid pain in production.

Buttons such as Save, Continue, Login, etc. should be disabled till form is not validated:

The entire application consisted of several forms which needed to be checked for appropriate validations and the buttons such as Save, Continue, Login, etc. needed to be enabled only after the forms are correctly validated. Implemented this functionality using AngularJS.

Fonts are not properly loaded in the Iframe:

This was a challenging task which dealt with browsers **CORS (Cross Origin Resource Sharing)** policy.

Convincely Identification Page used an **Iframe** within which the websites were loaded and CTAs (Call to Action) items were identified. The website displayed within this Iframe was not displaying the fonts for different text elements properly and the default browser font was being used due to which the look of the website within the iframe was ordinary. This was happening due to the browsers **CORS policy which did not allow loading of font files from the web server of the website**.

I designed a solution which identified all the **<link>** elements within the HTML page to find all links to style sheets on the web server. Then the style sheets would be fetched and parsed using the **cssutils python package** to look for **@font-face** rules. Then the **url** attribute of the **src** field of these font-face rules would give us the full path to the actual font files on the server. Then a **proxy url** through our own server was formed using this full path which provided a custom **Django API** that returned the content of the actual font file.

Then a **@font-face** rule similar to the one on the server but with the **proxy url** formed above would be inserted into a **<style>** tag into the **head** of the document. This solved the CORS problem and the website got loaded with proper good looking fonts into the Iframe.

Implementing the Remember Me Functionality for the Website:

The **Remember Me** feature required remembering the user session till he logs out.

Implemented this feature using the **AngularJS $cookies**. So whenever the user logs in with the **Remember Me** checkbox ticked, Convincely would set a cookie to store the token for the session with an expiry date of two weeks, so the cookie would persist till two weeks even if the browser is closed. This cookie would be destroyed on user logout action.

If the Remember Me checkbox is unticked while login, the token cookie would be set without an expiry date, which gets automatically destroyed on browser close.

Fixing the Back Button Functionality on page containing Iframe:

Pressing the Back button on a page containing iframe needed two back button clicks to go back to previous page.

This was happening because of wrong way of initializing the iframe which pushed an iframe state on the browser history stack, because of which the first back button click fired a back on iframe and then the next back button click fired a back on browser.

Fixed this by changing the way the iframe was initialized so that iframe state won't be pushed on history stack.

Implementing the functionality to log the user in after Demo is Complete:

Convincely allowed a **Guest User to Take a Free Tour** of the application as a guest. This allowed the Guest to enter the URL of his/her website and Start the Tour.

Once the Guest enters the URL and starts the tour, Identifications are made on the website for CTA (Call to Action) elements.

Once the user applies these Identifications by clicking the apply button, he would be taken to the Suggestions page where Suggestions would be displayed.

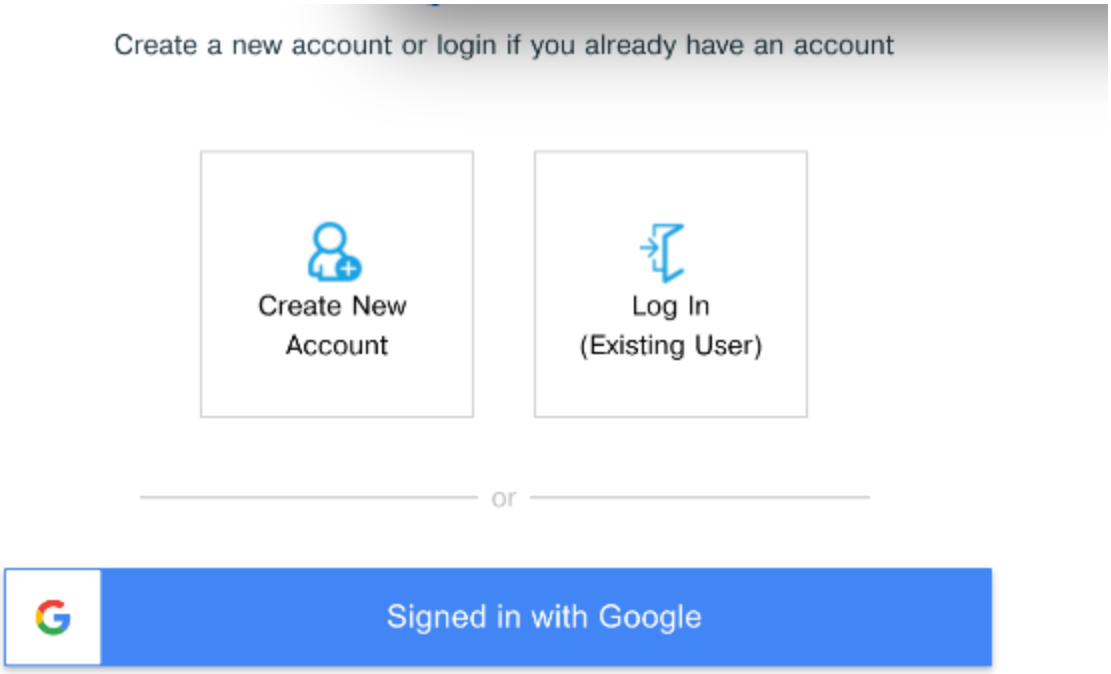
The demo is over once the user clicks Save Changes on the Suggestions page and he would be presented with a **dialog asking him to Register or Login**.

On successfully logging in from this phase, the user should be taken to the Create New Campaign page, with form data pre filled with information the User entered before starting the Tour.

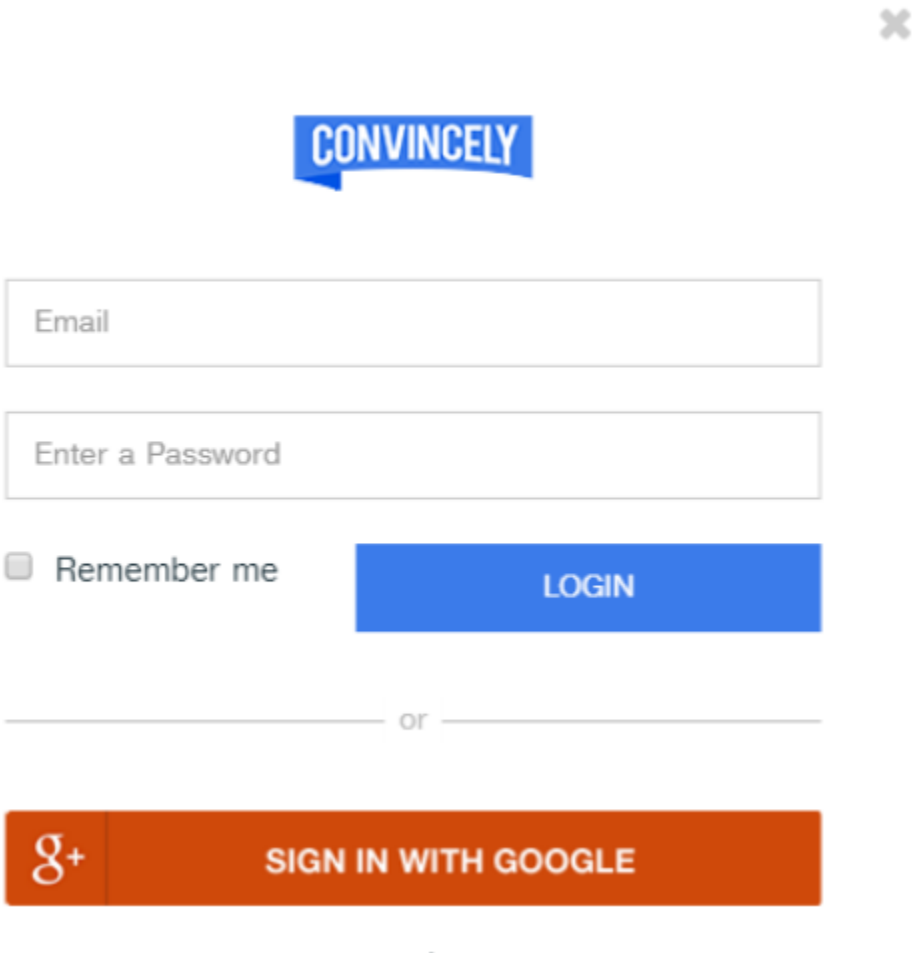
Implemented this functionality by adding code to the **Django REST API** for **Login** to create new Campaign objects from Guest Campaign objects in case the user logs in and then archiving the Guest Campaign objects.

Changing the design of the Google Sign In button:

Implementation of Google Sign In for Convincely led to a change in the look of the button to a blue colour button with a small Google sign on the left of it as shown below:



The UI/UX team had given a red colour button image to preserve the aesthetics of the website as shown below so that the user can differentiate between the Convincely sign in and Google Sign in:



This was happening because the default Google SignIn button belonged to **gsignin2** class. This class **coupled the design and the logic for gapi calls for login** together. To make a change to the design of the button to match the aesthetics of the website I had to remove this dependency on the **gsignin2** class.

Removed this dependency by manually implementing the gapi calls for login on button click and thus changed the design of the button using **custom CSS**.

Implementing Test Cases for the Django REST APIs:

Implemented test cases to test both positive and negative flows for each API using the Django test framework.

Personal Project:

Investment Diary:

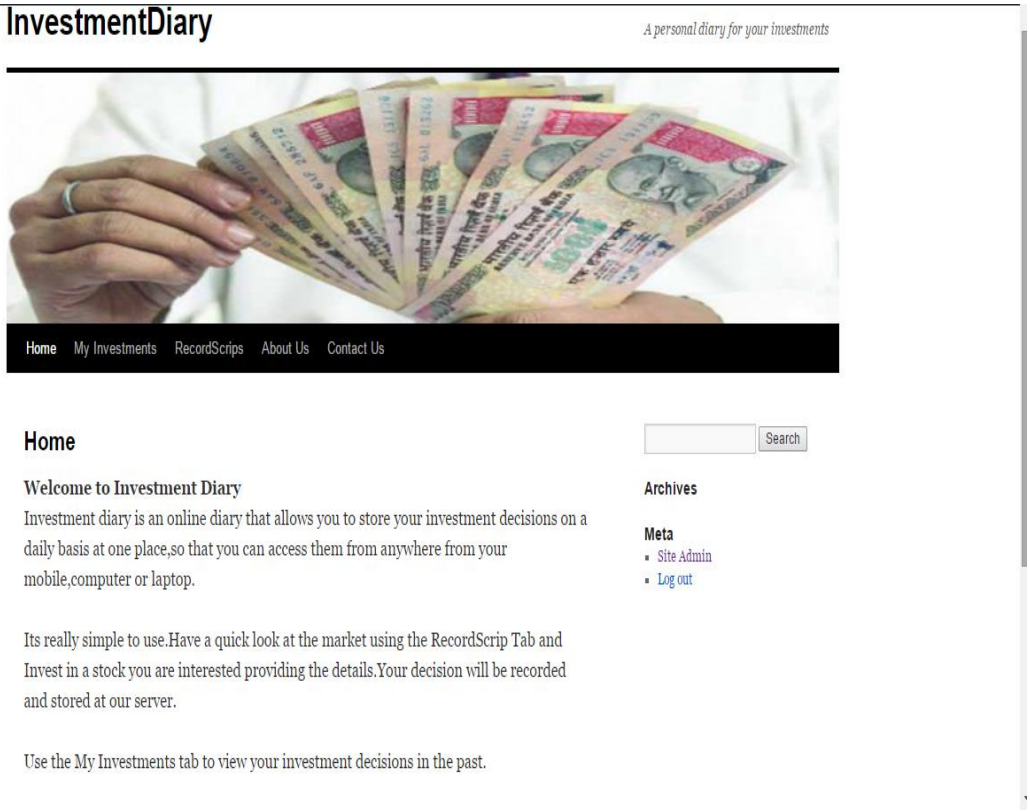
This project aimed at developing a website for recording one's stock market decisions at a certain point of time. Clients can use the website to view the current stock prices of various companies listed on the Bombay Stock Exchange. After looking at the prices and reading news about the future prospects of a particular company, if the client buys the shares of a particular company, he can record his transaction as a Record Scrip. The Record Scrip would store all the details of the transaction such as:

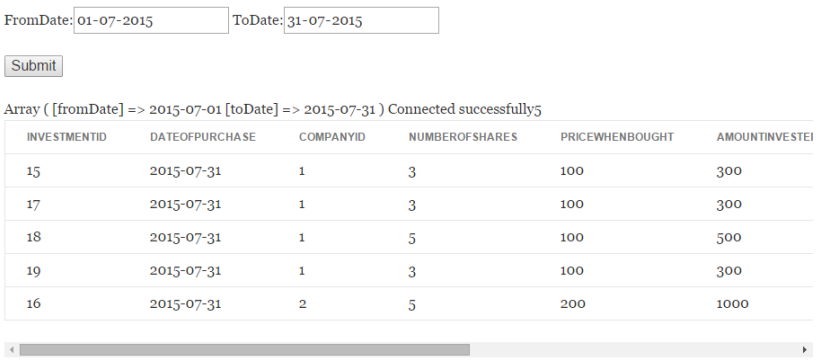
- The number of shares bought.
- Date of Purchase.
- Price at which the shares were bought.
- Additional charges per share.
- Total amount Invested.
- The target price (Price at which the client wants to sell).
- **Reason for buying the share.**

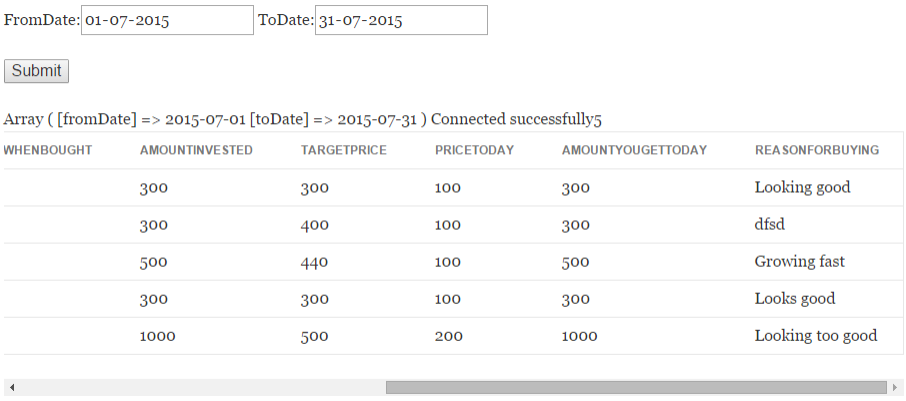
This data would be stored as a record at our server, and the client can access the data at any time by using the **My Investments** tab of the website and giving the date range.

We further plan to use the **Reason for buying the share,** field to perform in depth analysis about which shares are people investing in and why ,so that we can give our clients an overview of peoples sentiments for a particular company and also guide them in investing in a particular company.

I have attached some screenshots of the website below:





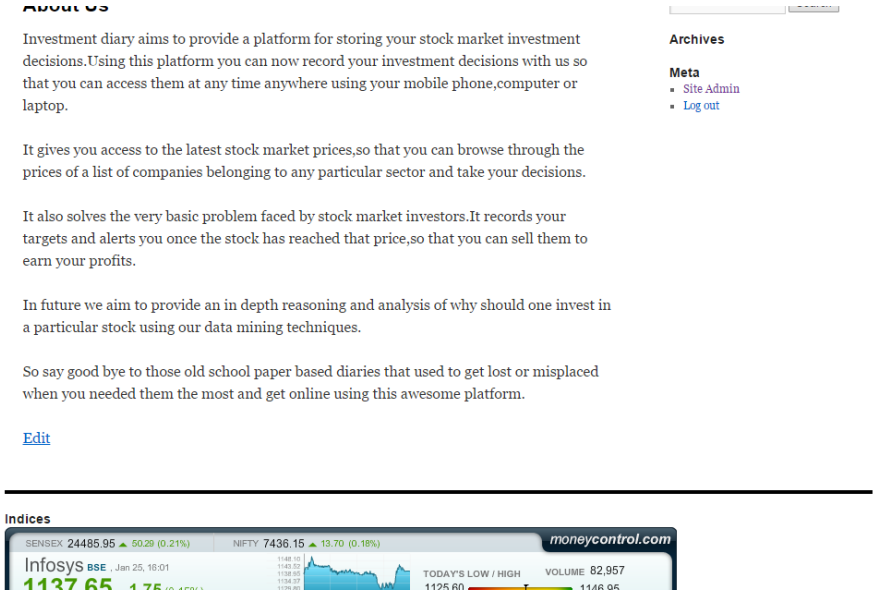


FromDate: 01-07-2015 ToDate: 31-07-2015

Submit

Array ([fromDate] => 2015-07-01 [toDate] => 2015-07-31) Connected successfully5

WHENBOUGHT	AMOUNTINVESTED	TARGETPRICE	PRICETODAY	AMOUNTYOUGETTODAY	REASONFORBUYING
	300	300	100	300	Looking good
	300	400	100	300	dfsd
	500	440	100	500	Growing fast
	300	300	100	300	Looks good
	1000	500	200	1000	Looking too good



Investment diary aims to provide a platform for storing your stock market investment decisions.Using this platform you can now record your investment decisions with us so that you can access them at any time anywhere using your mobile phone,computer or laptop.

It gives you access to the latest stock market prices,so that you can browse through the prices of a list of companies belonging to any particular sector and take your decisions.

It also solves the very basic problem faced by stock market investors.It records your targets and alerts you once the stock has reached that price,so that you can sell them to earn your profits.

In future we aim to provide an in depth reasoning and analysis of why should one invest in a particular stock using our data mining techniques.

So say good bye to those old school paper based diaries that used to get lost or misplaced when you needed them the most and get online using this awesome platform.

Edit

Archives

Meta

- Site Admin
- Log out

Indices

SENSEX 24485.95 ▲ 50.29 (0.21%) NIFTY 7436.15 ▲ 13.70 (0.18%) moneycontrol.com

Infosys BSE , Jan 25, 16:01

TODAY'S LOW / HIGH VOLUME 82,957

This was my friend's idea, and I helped him build some modules such as the back end database schema, the **Investment search** tab etc.

The website was built using Word press as the CMS.

My major involvement was in building the My Investments Tab to search investments using a date range and loading the data from the database without refreshing the webpage, using AJAX calls.

The website is still under construction and my friend is working on retrieving the Bombay Stock Exchange data in real time.

Seminar:

TE Seminar Topic: Use Of Nebulas in Cloud Computing.

Description: The seminar focused on the importance (both in terms of cost and efficiency) of a more dispersed cloud called a Nebula. It also included case studies of real implementations of Nebulas and compared the results with Centralized Cloud Computing.

Personal Details:

Full Name: SHAH VAIBHAV ANIL.

Nationality: Indian.

Birth date: 31st May 1992.

Place of Birth: Mumbai.

Gender: Male.

Marital Status: Single.

Valid Passport: Yes.
