INDIRA GANDHI NATIONAL OPEN UNIVERSITY

MCSP-060

Event Management System

By

Lakshay Verma

Enrolment No: 146433412

Under Guidance

Of

Mr. Prabal Kumar Joshi

Submitted to the School of Computer and Information Science, IGNOU

In partial fulfillment of the requirements

For the award of the degree

Masters of Computer applications (MCA)

INDIRA GANDHI NATIONAL OPEN UNIVERSITY

Maidan Ghari

New Delhi-110068

Index

Introduction/Objectives

An event management system is primarily used for management of all the activities related to event. In any event a lot of service providers work simultaneously and it gets very difficult to manage these providers. It is also very important for an event organizer that he has means to contact them any time in order to plan an event at given time. To manage all this I Lakshay Verma have developed this website. This website helps our user in getting success in the event management business. As a user should have strong network contacts of service provider. These contacts are essentially providers of specific services who can be mobilized quickly to participate in any given event. To make an event successful event manager needs different service provider like Sound systems services, Lighting providers, Canteen services, Stage construction/decorations and so on. In present system Event Company have to do all management work manually. They have to keep track of all items that will be present in an event, minor checklists here and there and guests list always at bay. Keeping all of these problems in mind I have developed this system. This system helps an event management company to manage their paper work online and keep track of multiple events in a single screen.

Event management market is growing at a rate of three times than that of traditional advertising. Though relatively small compared to the major components of the marketing communications mix-advertising, sales promotions and P-O-P communications-expenditures on event sponsorship are increasing. Corporate sponsorships in India in 2001 were estimated at $3.9 billion with 65% of this total going to sports events and most of the remainder spent on sponsoring entertainment tours or festival and fairs. Thousands of companies invest in some form of event sponsorship.

Defined, event marketing is a form of brand promotion that ties a brand with a meaningful athletic, entertaining, cultural, social or other type of high-interest public activity. Event marketing is distinct from advertising, sales promotion, point-of-purchase merchandising, or public relations, but it generally incorporates elements from all of these promotional tools. Event promotions have the opportunity to achieve success because, unlike other forms of marketing communications, events reach people when they are receptive to marketing messages and capture people in a relaxed atmosphere.

Event marketing is growing rapidly because it provides companies alternatives to the cluttered mass media, an ability to segment on a local or regional basis, and opportunities for reaching narrow lifestyle groups whose consumption behaviour can be linked with the local event. MasterCard invested an estimated $25 million in sponsoring the nine-city World Cup soccer championship in the United States in 1994 and will likely sponsor other big events in many countries as well.

Objective

Internet and Web technologies are growing day by day and today everyone is aware of this fact. No one has remained untouched from this invention and now our day is not complete without browsing through our favourite website whether that is *Social Media, Online shopping or Entertainment*. The **Event Management System** we are creating adheres to this fact and focuses on providing its end-users a one-stop portal for organising and managing multiple events at the same time. Also it helps to keep track of all the activities an event goes through during its life cycle.

Achievements

Event Management system has been developed with in a way that any number of persons can use the portal and start management of their own personal events in no time.

System Analysis

Every software project goes through a certain set of stages and processes that are standard for almost every company that has worked in the IT Industry. These stages are referred to as the SDLC or Software Development Life Cycle of the project. There are a lot of SDLC Models available to us for the process of software engineering. Some are quite simple and others are complicated enough to take months and months of practice to master them. For the current project, we have adopted the **'Spiral Model'** of software engineering.

Identification of Need

As mentioned earlier in the Objectives section of this report the current system is entirely based on paper based approach. Changes are being made to make the system digitized by keeping address books, mailing lists etc. But there still is room for improvement in such system. Which is why we have created a this **"Event Management System"** so users can easily perform the basic tasks and manage their events without need of hiring trained professionals for smaller events such as house parties and office meetings.

Also we have kept in mind that professional event managers also require a sophisticated system to improve their throughput in an event and that makes our system one of a kind to provide assistance to both trained professionals in the field of Event Management and the end users who have never done something like this before.

Preliminary Investigation

Primary investigation in the local area found that the concept of event management is not understood by many. They think of managing an event a tiresome task and neglect the fact that any event without proper management is a disaster in hiding. Some of the cases where a local event fail are listed below.

- Guests not arriving on time
- Improper chain of command
- Shortage of supplies
    - Not enough food
    - Improper lighting/sound
- No artists in an cultural event
- Unhappy guests
    - Because of sitting arrangements
    - Distasteful food and other eatables
- Under decorated venue

All of the above cases and others like them can be solved if the event managers and their team follow a system and we have tried to create such a system here so problems like this never happen as everyone keeps track of the tasks that are left and their status in between.

Feasibility Study

Creating any piece of software is a task that requires a lot of dedication and timeliness and creating an entire system takes more time than a basic software. It requires future planning before we even begin to start writing the first line of code. The very first task that we are required to perform is to conduct a feasibility study.

According to Wikipedia, **Feasibility Study** is an assessment of the practicality of a proposed project or system.

So we begin by finding if such a software can actually be used here under four categories.

Technical Feasibility

Determines whether the proposed system conflicts with legal requirements, e.g. a data processing system must comply with the local data protection regulations and if the proposed venture is acceptable in accordance to the laws of the land.

Our proposed System (website) runs on PHP stack and hence is open source will be easy to develop and launch as well. Which makes our system technically feasible.

Operational Feasibility

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases.

As discussed earlier our website will allow the user to perform all the basic event management tasks making our website operationally feasible.

Schedule Feasibility

A project will fail if it takes too long to be completed before it is useful. Typically this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Schedule feasibility is a measure of how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some projects are initiated with specific deadlines. It is necessary to determine whether the deadlines are mandatory or desirable.

The project of this website is required as the partial fulfilment of the MCA it needs to be on schedule and be submitted before 30th April 2017.

As per our calculations, this project is feasible in terms of schedule.

Project Planning

When you think of building a website, your thoughts rotate around two main issues – price and time. These two values depend largely on the size and scope of the project. To outline the whole development process, you can create a website development timeline, adding tasks and establishing milestones for your project. It is the best way to track your project implementation to make sure you keep up with the deadline.

We've prepared detailed description of the whole website development process, estimated time for each step and a checklist to double check you don't miss anything.

Despite conventional wisdom, the core part of website development and design is not necessary for the coding process. Indeed, such technologies as HTML, CSS, and JavaScript give the web we know its shape and define the way we interact with the information. But usually stay behind the scenes and, at the same time, remain the crucial part of website development life cycle are the stages of preliminary information gathering, detailed planning, and post-launch maintenance.

Spiral Model

The spiral model, originally proposed by Boehm, is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model. The spiral model is similar to the incremental model, with more emphasis placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed. Each subsequent spirals builds on the baseline spiral. It's one of the software development models like Waterfall, Agile, V-Model. A spiral model is divided into a number of framework activities, also called task regions. Typically, there are between three and six task regions. In below figure depicts a spiral model that contains six task regions:

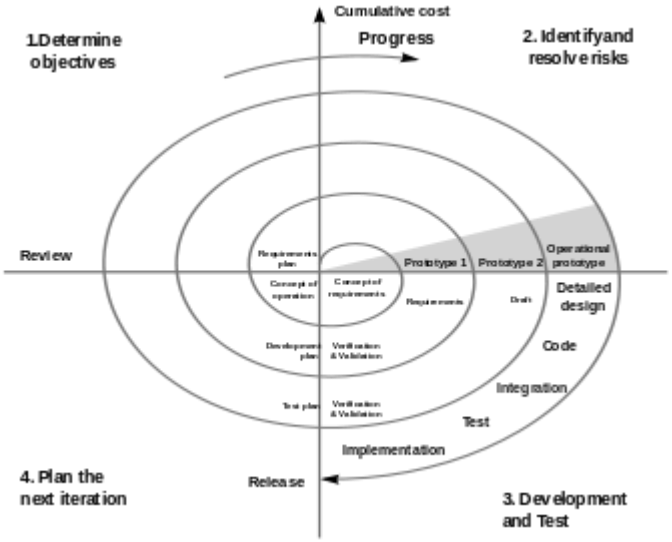


Figure 1 Spiral Model

Customer communication
—tasks required to establish effective communication between developer and customer.

Planning
—tasks required to define resources, timelines, and other project related information.

Risk analysis
—tasks required to assess both technical and management risks.

Engineering
—tasks required to build one or more representations of the application.

Construction and release
—tasks required to construct, test, install, and provide user support (e.g., documentation and training).

Customer evaluation
—tasks required to obtain customer feedback based on evaluation of the software representations that were created during the engineering stage and implemented during the installation stage.

Application of Spiral Model
The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e. learning with maturity which involves minimum risk for the customer as well as the development firms.

The following pointers explain the typical uses of a Spiral Model:

- When there is a budget constraint and risk evaluation is important.
- For medium to high-risk projects.
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.

- Customer is not sure of their requirements which is usually the case.
- Requirements are complex and need evaluation to get clarity.
- New product line which should be released in phases to get enough customer feedback.
- Significant changes are expected in the product during the development cycle.

Advantages of Spiral model:

- High amount of risk analysis hence, avoidance of Risk is enhanced.
- Good for large and mission-critical projects.
- Strong approval and documentation control.
- Additional Functionality can be added at a later date.
- Software is produced early in the software life cycle.

Disadvantages of Spiral model

- Can be a costly model to use.
- Risk analysis requires highly specific expertise.
- Project's success is highly dependent on the risk analysis phase.
- Doesn't work well for smaller projects.

When to use Spiral model

- When costs and risk evaluation is important
- For medium to high-risk projects
- Long-term project commitment unwise because of potential changes to economic priorities
- Users are unsure of their needs
- Requirements are complex
- New product line
- Significant changes are expected (research and exploration)

Conclusion

Each spiral can be termed as a loop and each loop is a separate development process in a spiral model. The four activities (Planning, Risk analysis, engineering and evaluation) form the intermediary phases of a spiral model and is repeated again for each loop.

This model is very good to use for larger projects where you can develop and deliver smaller prototypes and can enhance it to make the larger software. The implementation of this model requires experienced resources as risk analysis is a very integral part of this model and risk analysis requires expertise and as a result this model becomes costly.

We can draw four conclusions from the data presented:

(1) The risk-driven nature of the spiral model is more adaptable to the full range of software project situations than are the primarily document-driven approaches such as the waterfall model or the primarily code-driven approaches such as evolutionary development. It is particularly applicable to very large, complex, ambitious soft-wane systems.
(2) The spiral model has been quite successful in its largest application to date: the development and enhancement of the TRW-SPS. Overall, it achieved a high

level of software support environment capability in a very short time and provided the flexibility necessary to accommodate a high dynamic range of technical alternatives and user objectives.

(3) The spiral model is not yet as fully elaborated as the more established models. Therefore, the spinal model can be applied by experienced personnel, but it needs further elaboration in such areas as contracting, specifications, milestones, reviews, scheduling, status monitoring, and risk area identification to be fully usable in all situations.

Partial implementations of the spiral model, such as the risk management plan, are compatible with most current process models and are very helpful in overcoming major sources of project risk.

In this article, we'll take a look at how the general website development process may look like. The overall number of development stages usually varies from five to eight, but every time the whole picture stays pretty much the same. Let's choose the average value. So, here are seven main steps:

1. Information Gathering
2. Planning
3. Design
4. Content Writing and Assembly
5. Coding
6. Testing
7. Review
8. Launch
9. Maintenance

Step 1: Gathering Information

Purpose, Main Goals, and Target Audience

This stage, the stage of discovering and researching, determines how the subsequent steps will look like. The most important task at this point is to get the clear understanding of your future website purposes, the main goals you wish to get, and the target audience you want to attract to your site. Such kind of a website development questionnaire helps to develop the best strategy for further project management.

News portal differs from the entertainment websites, and online resources for teenagers looks different than sites for adults. Different types of websites provide visitors with different functionality which means that different technologies should be used according to the purposes. A well described and detailed plan made on the basis of this pre-development data can protect you from spending extra resources on solving the unexpected issues such as design changing or adding the functionality that wasn't initially planned.

Step 2: Planning

Sitemap and Wireframe Creation

At this stage of website development cycle, the developer creates the data that can give to a customer an opportunity to judge how the entire site will look like.

On the basis of the information that was gathered together in the previous phase, the sitemap is created. Here is the sitemap of our **Event Management** website:

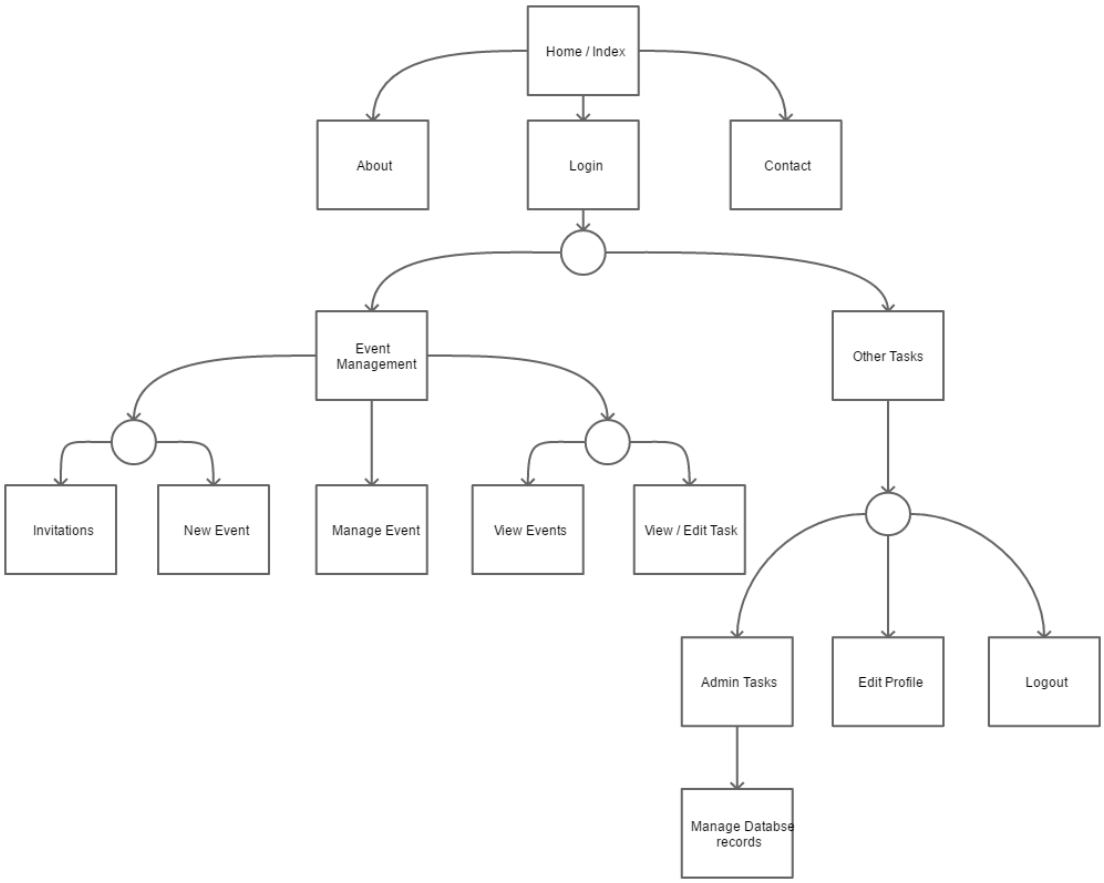


Figure 2 Site Map

The sitemap should describe the relations between the main areas of a website. Such representation could help understand how usable the final product will be. It can show you the *"relationship"* between the different pages of a website, so you can judge how easy it will be for the end-user to find the required information or service if he starts from the main page. The main reason behind the sitemap creation is to build a user-friendly and easy to navigate website.

The sitemap allows you to understand how the inner structure of a website looks like, but doesn't describe the user interface. Sometimes, before you start to code or even work on a design, there's a necessity to get approval from a customer that everything looks fine so you can begin the next phase of development. In this case, a ***wireframe or mock-up*** is created. A *wireframe* is a visual representation of user interface that you're going to create. But it doesn't contain any design elements such as colors, logos, etc. It only describes the elements that will be added to the page and their location. It's artless and cheap in production sketch.

You can use any mockup for this purpose. We used Pencil Project. Here's how the wireframe can look like:

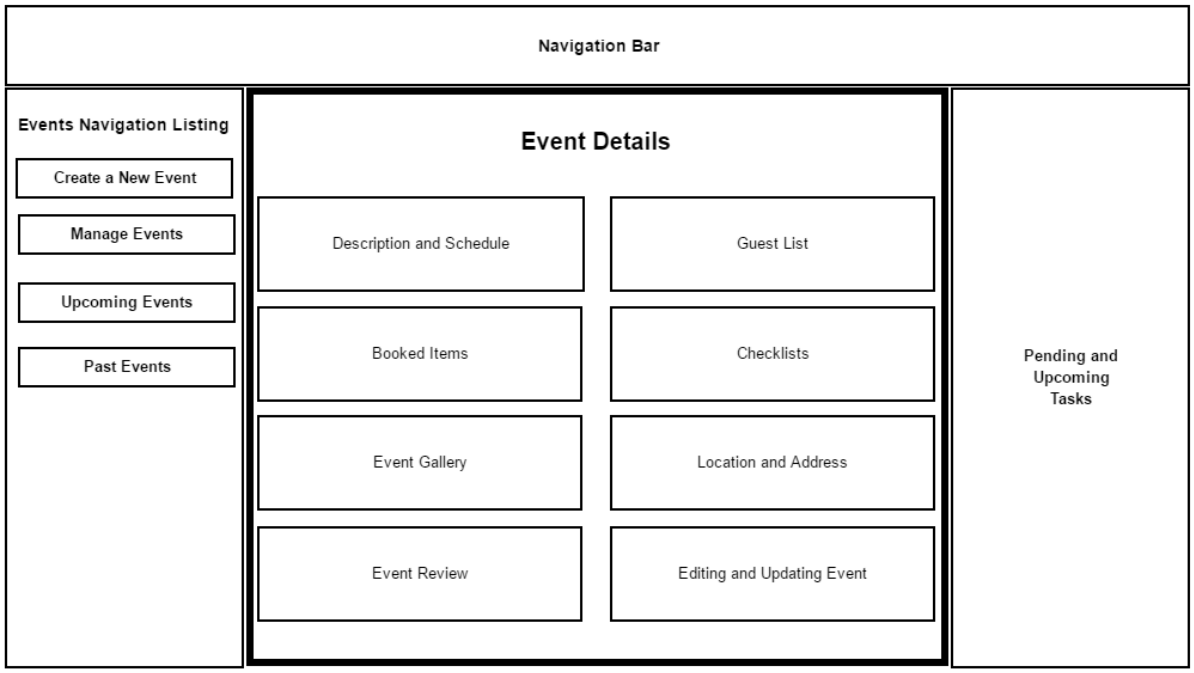


Figure 3 Wireframe

The other important thing is to select technology stack – programming language, frameworks, CMS that you're going to use.

Step 3: Design

Page Layouts, Review, and Approval Cycle

During the design phase, your website takes shape. All the visual content, such as images, photos, and videos is created at this step. Once again, all the info that was gathered through the first phase is crucial. The customer and target audience must be kept in mind while you work on a design.

Website layout is the result of designer's work. It can be a graphic sketch or an actual graphic design. The primary function of the layout is to represent the information structure, visualize the content, and demonstrate the basic functional. Layouts contain colors, logos, images and can give a general understanding of the future product.

After that, the customer can review the layout and send you his feedback. If the client is not sure about some aspects of your design, you should change the layout and send it back to him. This cycle should be repeated until the customer is completely satisfied.

Step 4: Content Writing and Assembly

Content writing and compiling usually overlaps with other stages of website creation, and its role can't be underestimated. At this step it is necessary to put in writing the very essence you'd like to communicate to the audience of your website and add calls-to-action. Content writing involves also creation of catching headlines, text editing, writing new text, compiling the existing text, etc., which takes time and effort. As a rule, the client undertakes to provide website content ready to migrate to the site. It is better when all website content is provided before or during website coding.

Step 5: Coding

At this step, you can finally start creating the website itself. Graphic elements that have been designed during the previous stages should be used to create an actual website. Usually, the home page is created first, and then all sub-pages are added, according to the website hierarchy that was previously created in the form of a sitemap. Frameworks and CMS should be implemented to make sure that server can handle the installation and set-up smoothly.

All static web page elements that were designed during the mock-up and layout creation should be created and tested. Then, special features and interactivity should be added. A deep understanding of every website development technology that you're going to use is crucial at this phase.

When you use CMS for site creation, you can also install CMS plugins at this step if there's a need. The other important step is SEO (Search Engine Optimization). SEO is the optimization of website elements (e.g., title, description, and keyword) that can help your site achieve higher rankings in the search engines. And, once again, valid code is pretty important for SEO.

Step 6: Testing, Review and Launch

Testing is probably the most routine part of a process. Every single link should be tested to make sure that there are no broken bones among them. You should check every form, every script, run a spell-checking software to find possible typos. Use code validators to check if your code follows the current web standards. Valid code is necessary, for example, if cross-browser compatibility is important for you.

After you check and re-check your website, it's time to upload it to a server. An FTP (File Transfer Protocol) software is used for that purpose. After you deployed the files, you should run yet another, final test to be sure that all your files have been installed correctly.

Step 7: Maintenance

Opinion Monitoring and Regular Updating

What's important to remember is that a website is more a service than a product. It's not enough to "deliver" a website to a user. You should also make sure that everything works fine, and everybody is satisfied and always be prepared to make changes in another case.

Feedback system added to the site will allow you to detect possible problems the end-users face. The highest priority task in this case is to fix the problem as fast as you can. If you won't, you may find one day that your users prefer to use another website rather than put up with the inconvenience.

The other important thing is keeping your website up to date. If you use a CMS, regular updates will prevent you from bugs and decrease security risks.

Conclusion

You should always keep in mind that website development project doesn't start with coding and doesn't end after the day you finally launch your website. The phase of preparation affects all subsequent stages, defining how productive the development

process will be. A profound and deep discovery of such aspects like age, sex, and interests of your end-user may become the key to success. The post-launch period is rather significant. Your project should be agile and flexible enough to have a possibility to change your website according to users' feedback or spirit of the time. Keeping in mind that there's no such thing as insignificant website development phase will prevent you from unexpected troubles and give you confidence that everything flows as it should, and you have full control over the project.

Project Scheduling

The project schedule is the tool that communicates what work needs to be performed, which resources of the organization will perform the work and the timeframes in which that work needs to be performed. The project schedule should reflect all of the work associated with delivering the project on time. Without a full and complete schedule, the project manager will be unable to communicate the complete effort, in terms of cost and resources, necessary to deliver the project.

Online project management software allows project managers to track project schedules, resources, budgets and project related assets in real time. The project schedule can be viewed and updated by team members associated with the project, keeping everyone well informed on the overall project status.

Methods

Before a project schedule can be created, the schedule maker should have a work breakdown structure (WBS), an effort estimate for each task, and a resource list with availability for-each resource. If these components for the schedule are not available, they can be created with a consensus-driven estimation method like Wideband Delphi. The reason for this is that a schedule itself is an estimate: each date in the schedule is estimated, and if those dates do not have the buy-in of the people who are going to do the work, the schedule will be inaccurate.

In order for a project schedule to be healthy, the following criteria must be met

The schedule must be constantly (weekly works best) updated.

The EAC (Estimation at Completion) value must be equal to the baseline value.

The remaining effort must be appropriately distributed among team members (taking vacations into consideration).

The schedule structure may closely follow and include citations to the index of work breakdown structure or deliverables, using decomposition or templates to describe the activities needed to produce the deliverables defined in the WBS.

A schedule may be assessed for the quality of the schedule development and the quality of the schedule management.

Work Breakdown Structure (WBS)

The building blocks of a schedule start with a Work Breakdown Structure (WBS). The WBS is a hierarchical reflection of all the work in the project in terms of deliverables. In order to produce these deliverables, work must be performed.

A typical approach in developing a WBS is to start at the highest level, with the product of the project. For example, you are assigned as the project manager of a New Product Development project. The new product you are developing is a new toy for children age's five through nine. The objective of this product development project is to increase the revenue of the organization by ten percent.

WBS is a hierarchical and incremental decomposition of the project into phases, deliverables and work packages. It is a tree structure, which shows a subdivision of effort required to achieve an objective; for example a program, project, and contract. In a project or contract, the WBS is developed by starting with the end objective and successively subdividing it into manageable components in terms of size, duration, and responsibility (e.g., systems, subsystems, components, tasks, subtasks, and work packages) which include all steps necessary to achieve the objective.

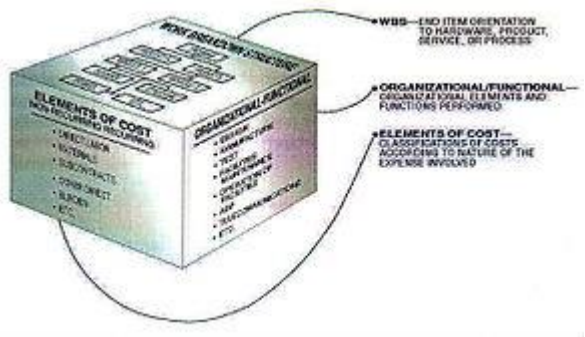


Figure 4 Example of work breakdown structure applied in a NASA reporting structure

The work breakdown structure provides a common framework for the natural development of the overall planning and control of a contract and is the basis for dividing work into definable increments from which the statement of work can be developed and technical, schedule, cost, and labour hour reporting can be established.

A work breakdown structure permits summing of subordinate costs for tasks, materials, etc., into their successively higher level "parent" tasks, materials, etc. For each element of the work breakdown structure, a description of the task to be performed is generated. This technique (sometimes called a system breakdown structure) is used to define and organize the total scope of a project.

The WBS is organized around the primary products of the project (or planned outcomes) instead of the work needed to produce the products (planned actions). Since the planned outcomes are the desired ends of the project, they form a relatively stable set of categories in which the costs of the planned actions needed to achieve them can be collected. A well-designed WBS makes it easy to assign each project activity to one and only one terminal element of the WBS. In addition to its function in cost accounting, the WBS also helps map requirements from one level of system

specification to another, for example a requirements cross reference matrix mapping functional requirements to high level or low level design documents. The WBS may be displayed horizontally in outline form, or vertically as a tree structure (like an organization chart).

The development of the WBS normally occurs at the start of a project and precedes detailed project and task planning.

For this project we can refer to our Site Map in order to create an organised WBS.

Gantt chart

A Gantt chart is a type of bar chart, devised by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities.

For creating a Gantt chart we have used a free software Gantt Project and using the same the following chart was created.

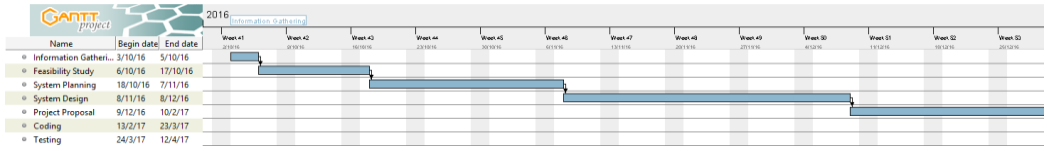


Figure 5 Project Gantt Chart 1

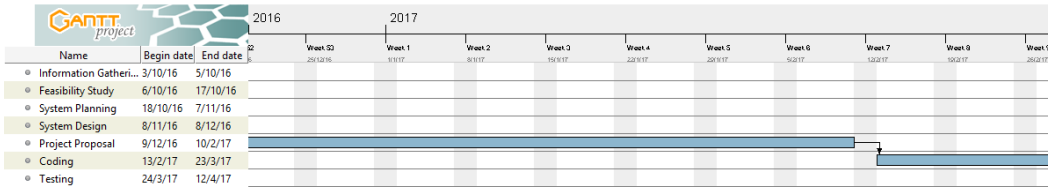


Figure 6 Project Gantt Chart 2

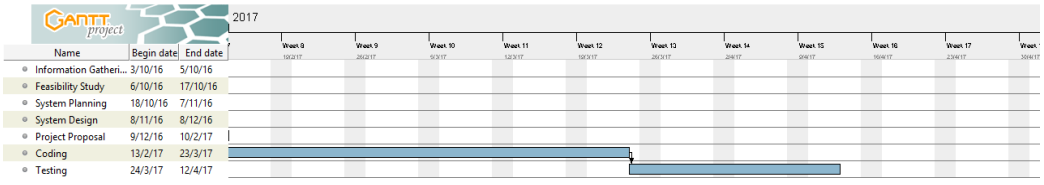


Figure 7 Project Gantt Chart 3

PERT Chart

Program Evaluation and Review Technique represents all task that are required for project's completion along with their corresponding time requirements. For a quality software, in our case a website, we require it to follow the protocols and sometimes we even create Gantt Charts to divide workload of our project into manageable time slots. A PERT chart presents a graphic illustration of a project as a network diagram

consisting of numbered nodes (either circles or rectangles) representing events, or milestones in the project linked by labelled vectors (directional lines) representing tasks in the project. The direction of the arrows on the lines indicates the sequence of tasks.

PERT is a method of analyzing the tasks involved in completing a given project, especially the time needed to complete each task, and to identify the minimum time needed to complete the total project.

PERT was developed primarily to simplify the planning and scheduling of large and complex projects. It was developed for the U.S. Navy Special Projects Office in 1957 to support the U.S. Navy's Polaris nuclear submarine project. It was able to incorporate uncertainty by making it possible to schedule a project while not knowing precisely the details and durations of all the activities. It is more of an event-oriented technique rather than start- and completion-oriented, and is used more in projects where time is the major factor rather than cost. It is applied to very large-scale, one-time, complex, non-routine infrastructure and Research and Development projects. An example of this was for the 1968 Winter Olympics in Grenoble which applied PERT from 1965 until the opening of the 1968 Games.

This project model was the first of its kind, a revival for scientific management, founded by Frederick Taylor (Taylorism) and later refined by Henry Ford (Fordism). DuPont's critical path method was invented at roughly the same time as PERT.

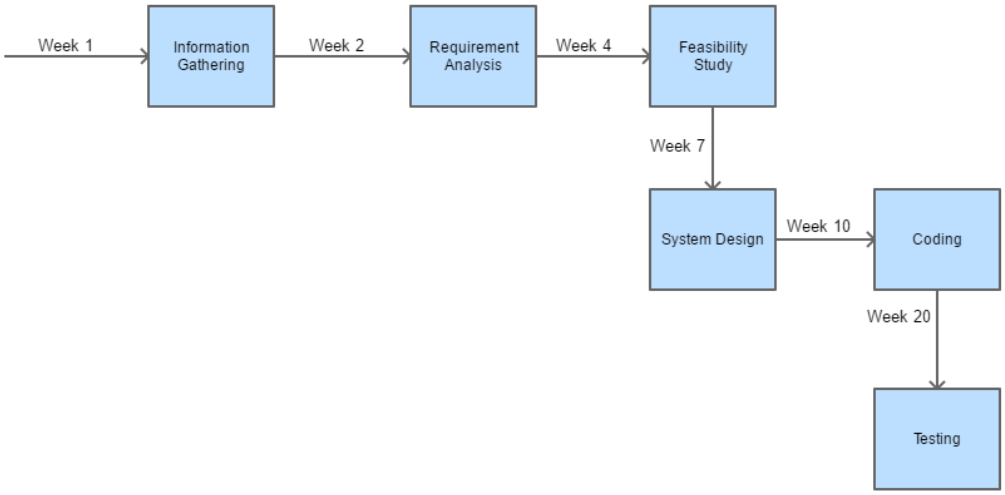


Figure 8 Pert Chart

Software requirement specification

Summary of Event Management

Event Management generally refers to any service provided in relation to planning, promotion, organizing or presentation of any arts, entertainment, business, sports or any other event and includes any consultation provided in this regard. The current condition of the event management companies in the city is not very well as they are waiting for the response of the people so that in future they can encash it. In fact right now the role of event management companies is being played by either the hotels or the amusement parks or the community centres. In many cases newspapers work in collaboration with private parties to organize some event. The findings of the project clearly states that the future of event management is very bright in the city although presently it's in nascent stage but the people in our National capital Delhi have started accepting such events in the main line. Some such events are like theme based marriages and "Ladies Sangeet" function the marriages also people now a days are inclined towards destination weddings that requires a lot of pre planning. Along with this the growth in gathering of fashion shows and musical concerts is also remarkable. Right now the condition of event management in our country is not that good and the response of people has not been positive to an extent that it can attract any event management company here in India as a full-fledged service provider, it will take some time to pick up the pace nicely. Currently the main target segment of these partial or complete service providers are the defence class people or the high society and rich people as they have accepted these concepts much widely than any other community. It is expected that event management industry will gear up in the near future. The concept needs to be popularized, by proper media coverage. The existing companies should mould their service according to the taste and preference of the people. As per our findings and the calculations the results are not very encouraging at present. But the projections help us to draw a conclusion that there is a scope for event management companies in near future.

Meaning of "Event Management"

Meaning of the word 'EVENT'

An event is a live multimedia package carried out with preconceived concept, customized or modified to achieve the clients' objectives of reaching out and suitably influencing the sharply defined, specially gathered target audience by providing a complete experience and an avenue for two way interaction.

> *"The use of unconventional media / method involving people witnessing a happening within a capsule of time, for the purpose of communication of a message."*

Event in terms of Cultural and social life

In cultural and social life, an event refers to a social gathering or activity, such as:

- **Festivals**, for example a musical festival

• **Ceremonies**, for example a marriage

• **Parties**, for example a birthday party

Event in terms of science
In science, an event is something that takes place at a particular place and time. Specifically, event may refer to:

• Phenomenon, something observable at a given time

• A point in space time, a concept of the theory of relativity

• Event (particle physics), a set of elementary particle interactions

• Event (probability theory), a possible outcome of an experiment

• Extinction event, when a large number of biological species die out in a relatively short period of time.

Programming
In programming, an event is a software message that indicates something has happened, such as a keystroke or mouse click.

Common usage
In common usage, an event (as opposed to a special event) has a connotation of anocc urrence which is more common than a phenomenon (due perhaps to the difference between a two syllable word and a four syllable word). Thus, in common usage, a keystroke is an event, where the Big Bang might be connoted a phenomenon (a special event which denotes the beginning of the universe). For example, portal: current events denotes “events” rather than “phenomena”.

Meaning of the Word ‘Management’

The term "management" characterizes the process of and the personnel leading and directing all or part of an organization through the deployment and manipulation of resources (human, financial, material, intellectual or intangible).According to the Oxford English Dictionary, the word "manage" comes from the Italian maneggiare (to handle). Management has to do with power by position, whereas leadership involves power by influence

Type of Events

These days as said earlier, event management services are in high demand in some area, as more events are being arranged in increasing numbers. Over the past ten to fifteen years, the event management industry has witnessed huge growth. Each year, across the world, nearly $500 billion is spent on planned events.

International Event management companies are hired to organise a wide range of events. These companies can organise events for small groups of people, or big events with thousands of attendees. Lots of businesses use event management services, to ensure that their events run smoothly and professionally. Event planning is a stressful, time consuming and expensive activity. An event management company has the expertise and industry contacts to offer a dependable service, at an affordable price.

Let's take a closer look at how these companies approach organising different types of events:

Leisure Events

Leisure event management is an interesting and diverse field. It involves managing facilities like sports grounds, recreation centres, parks and entertainment venues. Also, it can involve managing a celebration or festival, sporting contest or concert.

The leisure events companies that manage these activities hire an engaged and enthusiastic workforce, who know their target audience. Moreover, these professionals have the expertise to carry out all the necessary administration and planning. The skill of the leisure event manager lies in his/her ability to understand what the event taking place means to those who are in attendance.

Also, a leisure event manager needs to incorporate value adding measures, to improve the experience of event goers. These measures could be delivered via technology, such as simulators or white knuckle rides in visitor attractions, or screen and sound systems in cinemas. The measures could be delivered through staff management, with service blueprinting to define staff roles in animating, motivating and engaging with customers. Or, the measures could be delivered through other work procedures and customer communication methods, to enhance the service quality, customer flow and circulation. This might encompass ticket purchasing, automatic entry, sight lines and signage, booking systems, and other similar technology that underpins the structure of an event.

Cultural Events

Often, cultural events are meant to enrich the cultural standing of the city in which they are held. Cultural events teams create unique festivals, memorable outdoor spectacles, accessible entertainment and unexpected arts. Often, these teams work alongside local government authorities. Every year, they will plan innovative cultural programmes and events, and advertise and oversee them from start to finish.

Managing a cultural event is a major undertaking. Festivals seem glamorous and exciting but in reality, they are hard work in every respect. Cultural event managers have to deliver an artistic and cultural programme, which will appeal to the public. However, they have to take local sensitivities into account, adhere to strict planning and health and safety rules, and still make a profit.

Cultural event managers need skills in fundraising and finance, public relations and arts marketing. Typically, these events management companies have tight budgets, and are dependent on part time staff, enthusiastic volunteers and freelance contractors. Obviously, styles of cultural events management vary, based on the genre and size of events. Nonetheless, all cultural events are linked to the well-being of the community and issues of identity. Also, they all tap into the visitor and tourist economy.

Personal Events

Invariably, it takes a lot of time, effort and dedication to organise a personal event. With catering arrangements, types of menus, guest lists, booking the venue and designing the décor, there are countless details that have to be scheduled and coordinated. In the past, personal events management companies only dealt with indoor events. However, nowadays, these companies manage outdoor events as well.

A personal event management company will bring a party to life, with colours, patterns, spectacular backdrops and lighting effects that will enchant all the guests present. Indeed, these companies specialise in organising funky, uniquely themed events. The company will consult with the customer to discuss the precise requirements, and then tailor the event in line with the customer's budget. Venue booking, marquee erecting, menu planning and live entertainment provision is all included in the service.

Not all customers for these types of events require complete event management. Hence, personal event management services can be enlisted to organise only part of an event — for example, the hiring of entertainment or the catering. Initial consultations will determine which parts of the event would be best left to a professional organiser, and which parts the event hosts can tackle themselves. Often, the division of labour is decided by budget, and by the customer's relative expertise.

Organisational Events

Organisational events can include political, charitable and commercial events, as well as sales events, such as product launches, etc. A company that hosts an all-day event for several thousand people will require catering, entertainment and accommodation arranging for all the guests. Event staff will need to be recruited, a room to host the event will have to be chosen, seating arrangements will need to be determined and obviously, an event budget has to be established. An organisational events management company will have expertise in all of these areas.

Also, these companies might be hired to do more than simply plan and execute an event. They may be enlisted to implement a communications, marketing and brand building strategy as well. An organisational events manager understands the technical, logistical and creative factors that make an event successful. This includes audio visual production, event design, logistics, scriptwriting, negotiation, budgeting and of course, customer service. For bigger organisational events, in the preparation stage, the events manager has to make important decisions and choices linked to the event's creative concept and design. To ensure that the event runs to plan, the events manager requires extensive technical design expertise, and a clear grasp of how to convey a company's message in the public arena.

Obviously, event management is a multifaceted profession. The scale of the task involved in managing the above types of events is significant. It is hardly surprising that the industry is thriving.

Purpose

Event management system responds to its users who might be Organising an Event or attending it. This system allows an *Event Organiser* to keep birds eye view on matters such as who is attending the event who is not, which items have been booked for the event, who is assigned what task, etc. On the other hand a *Guest* can look at the details of same event such as Location, Time, Date, other guests and after event concludes they can rate the same and view gallery of photographs that were taken during the course of event.

Overall Description

Functions of management

Management operates through various functions, often classified as planning, organizing, leading/motivating and controlling.

1. **Planning**: Deciding what has to happen in the future (today, next week, next month, next year, over the next five years, etc.) and generating plans for action.
2. **Organizing**: Making optimum use of the resources required to enable the successful carrying out of plans.
3. **Leading/Motivating**: Exhibiting skills in these areas for getting others to play an effective part in achieving plans.
4. **Controlling/Monitoring**: checking progress against plans, which may need modifications based on feedback.
5. **Staffing**: Appointing skill and unskilled workers, and efficient personnel.

Scope

1. Information about an event like its date, time, and location is on separate papers and might get misplaced by human error.
2. Having no chain of command during the management of event and even afterwards.
3. Guests do not know if to go to an event they know nothing about.
4. Event organiser to get estimates about how many guests will be attending/coming to the event.
5. Planning event's schedule is tiresome if done on paper as we might miss one detail thus putting event in jeopardy.

The web-site for Event Management solves all the issues thus making it better than the paper based approach.

Applicability

The website is developed using PHP and MySQL as the major components to make is easily accessible and super easy to implement by using an Apache, MySQL and PHP stack such as WAMP or XAMP for Windows machines.

 # For Apple Mac we have XAMP and for Linux machines, we can use either LAMP or XAMP.

Interfaces

Software Interfaces

The project is web based that is why we require a web server that responds to all the requests made by client. For making this project feasible and easy to develop by following the industry standards we can use the following technologies.

Apache

The Apache Software Foundation (ASF) is an American non-profit corporation to support Apache software projects, including the Apache HTTP Server. The ASF was formed from the Apache Group and incorporated in Delaware, U.S., in June 1999.

Figure 9 Apache Server Foundation

The Apache Software Foundation is a decentralized open source community of developers. The software they produce is distributed under the terms of the Apache License and is free and open source software (FOSS). The Apache projects are characterized by a collaborative, consensus-based development process and an open and pragmatic software license. Each project is managed by a self-selected team of technical experts who are active contributors to the project. The ASF is a meritocracy, implying that membership of the foundation is granted only to volunteers who have actively contributed to Apache projects. The ASF is considered a second generation open-source organization, in that commercial support is provided without the risk of platform lock-in.

Among the ASF's objectives are: to provide legal protection to volunteers working on Apache projects; to prevent the Apache brand name from being used by other organizations without permission. Installing Apache on Linux does require a bit of programming skills (though it is not too difficult). Installing it on a Windows platform is straight forward, as you can run it through a graphical user interface.

Apache's original core is fairly basic and contains a limited number of features. Its power rather comes from added functionality introduced through many modules that are written by programmers and can be installed to extend the server's capabilities. To add a new module, all you need to do is install it and restart the Apache server. Functionality that you don't need or want can easily be removed which is actually considered a good practice as it keeps the server small and light, starts faster, consumes less system resources and memory, and makes the server less prone to security holes. The Apache server also supports third party modules, some of which have been added to Apache 2 as permanent features. The Apache server very easily integrates with other open source applications, such as PHP and MySQL, making it even more powerful than it already is.

A web server in its simplest form is a computer with special software, and an internet connection that allows it to connect to other devices.

Every device connected to a network has an IP address through which others connect to and communicate with it. This IP address is sort of like a regular address that you need in real life to call or visit any contact of yours. If they didn't have an address, you wouldn't know how to call or reach them. IP addresses serve the exact same purpose. If a device didn't have one, the other machines on the same network wouldn't know how to reach it.

The Apache server offers a number of services that clients might make use of. These services are offered using various protocols through different ports, and include: hypertext transfer protocol (HTTP), typically through port 80, simple mail transfer protocol (SMTP), typically through port 25, domain name service (DNS) for mapping domain names to their corresponding IP addresses, generally through port 53, and file transfer protocol (FTP) for uploading and downloading files, usually through port 21.

How Apache Works

Apache's main role is all about communication over networks, and it uses the TCP/IP protocol (Transmission Control Protocol/Internet Protocol which allows devices with IP addresses within the same network to communicate with one another).

The TCP/IP protocol is a set of rules that define how clients make requests and how servers respond, and determine how data is transmitted, delivered, received, and acknowledged.

The Apache server is set up to run through configuration files, in which directives are added to control its behavior. In its idle state, Apache listens to the IP addresses identified in its config file (HTTPd.conf). Whenever it receives a request, it analyzes the headers, applies the rules specified for it in the Config file, and takes action.

But one server can host many websites, not just one - though, to the outside world, they seem separate from one another. To achieve this, every one of those websites has to be assigned a different name, even if those all map eventually to the same machine. This is accomplished by using what is known as virtual hosts.

Since IP addresses are difficult to remember, we, as visitors to specific sites, usually type in their respective domain names into the URL address box on our browsers. The browser then connects to a DNS server, which translates the domain names to their IP addresses. The browser then takes the returned IP address and connects to it. The browser also sends a Host header with the request so that, if the server is hosting multiple sites, it will know which one to serve back.

For example, typing in www.google.com into your browser's address field might send the following request to the server at that IP address:

```
1       GET / HTTP/1.1
2       Host: www.google.com
```

The first line contains several pieces of information. First, there is the method (in this case it's a GET), the URI, which specifies which page to be retrieved or which program to be run (in this case it's the root directory denoted by the /), and finally there is the HTTP version (which in this case is HTTP 1.1).

HTTP is a request / response stateless protocol.

HTTP is a request / response stateless protocol. It's a set of rules that govern communication between a client and the server. The client (usually but not necessarily a web browser) makes a request, the server sends back a response, and communication stops. The server doesn't look forward for more communication as is the case with other protocols that stay at a waiting state after the request is over.

If the request is successful, the server returns a 200 status code (which means that the page is found), response headers, along with the requested data. The response header of an Apache server might look something like the following:

```
01      HTTP/1.1 200 OK
02      Date: Sun, 10 Jun 2012 19:19:21 GMT
03      Server: Apache
04      Expires: Wed, 11 Jan 1984 05:00:00 GMT
05      Cache-Control: no-cache, must-revalidate, max-age=0
06      Pragma: no-cache
07      Last-Modified: Sun, 10 Jun 2012 19:19:21 GMT
08      Vary: Accept-Encoding,User-Agent
09      Content-Type: text/html; charset=UTF-8
10      Content-Length: 7560
```

The first line in the response header is the status line. It contains the HTTP version and the status code. The date follows next, and then some information about the host server and the retrieved data. The Content-Type header lets the client know the type of data retrieved so it knows how to handle it. Content-Length lets the client know the size of the response body. If the request didn't go throw, the client would get an error code and message, such as the following response header in case of a page not found error:

```
1       HTTP/1.1 404 Not Found
```

PHP

PHP (recursive acronym for PHP: Hypertext Pre-processor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

Figure 10 PHP

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

PHP code may be embedded into HTML or HTML5 markup, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification.

The PHP interpreter only executes PHP code within its delimiters. Anything outside its delimiters is not processed by PHP, although non-PHP text is still subject to control structures described in PHP code. The most common delimiters are <?php to open and ?> to close PHP sections. The shortened form <? also exists. This short delimiter makes script files less portable, since support for them can be disabled in the local PHP configuration and it is therefore discouraged. However, there is no recommendation against the use of the echo short tag <?=. Prior to PHP 5.4.0, this short syntax for echo() only works with the short_open_tag configuration setting enabled, while for PHP 5.4.0 and later it is always available. The purpose of all these delimiters is to separate PHP code from non-PHP content, such as JavaScript code or HTML markup.

The first form of delimiters, <?php and ?>, in XHTML and other XML documents, creates correctly formed XML processing instructions. This means that the resulting mixture of PHP code and other mark-up in the server-side file is itself well-formed XML.

Variables are prefixed with a dollar symbol, and a type does not need to be specified in advance. PHP 5 introduced type hinting that allows functions to force their parameters to be objects of a specific class, arrays, interfaces or call back functions. However, before PHP 7.0, type hints could not be used with scalar types such as integer or string.

Unlike function and class names, variable names are case sensitive. Both double-quoted ("") and here doc strings provide the ability to interpolate a variable's value into the string. PHP treats newlines as whitespace in the manner of a free-form language, and statements are terminated by a semicolon. PHP has three types of comment syntax: /* */ marks block and inline comments; // as well as # are used for one-line comments. The echo statement is one of several facilities PHP provides to output text, e.g., to a web browser.

In terms of keywords and language syntax, PHP is similar to the C style syntax. if conditions, for and while loops, and function returns are similar in syntax to languages such as C, C++, C#, Java and Perl.

Data types

PHP stores integers in a platform-dependent range, either a 64-bit or 32-bit signed integer equivalent to the C-language long type. Unsigned integers are converted to signed values in certain situations; this behaviour is different from that of other programming languages.] Integer variables can be assigned using decimal (positive and negative), octal, hexadecimal, and binary notations.

Floating point numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of scientific notation.] PHP has a native Boolean type that is similar to the native Boolean types in Java and C++. Using the Boolean type conversion rules, non-zero values are interpreted as true and zero as false, as in Perl and C++.

The null data type represents a variable that has no value; NULL is the only allowed value for this data type.

Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension; examples include file, image, and database resources.

Arrays can contain elements of any type that PHP can handle, including resources, objects, and other arrays. Order is preserved in lists of values and in hashes with both keys and values, and the two can be intermingled. PHP also supports strings, which can be used with single quotes, double quotes, nowdoc or heredoc syntax.

The Standard PHP Library (SPL) attempts to solve standard problems and implements efficient data access interfaces and classes.

Functions

PHP defines a large array of functions in the core language and many are also available in various extensions; these functions are well documented in the online PHP

documentation. However, the built-in library has a wide variety of naming conventions and associated inconsistencies, as described under history above.

Custom functions may be defined by the developer, e.g.:

```
function myAge($birthYear) {
// defines a function, this one is named "myAge"
    $yearsOld = date('Y') - $birthYear;
// calculates the age
    return $yearsOld . ' year' . ($yearsOld != 1 ? 's' :
''); // returns the age in a descriptive form
}

echo 'I am currently ' . myAge(1981) . ' old.';
// outputs the text concatenated

// with the return value of myAge()
// As the result of this syntax, myAge() is called.
```

In 2017, the output of the above sample program is 'I am currently 36 years old.'

In lieu of function pointers, functions in PHP can be referenced by a string containing their name. In this manner, normal PHP functions can be used, for example, as callbacks or within function tables.] User-defined functions may be created at any time without being prototyped. Functions may be defined inside code blocks, permitting a run-time decision as to whether or not a function should be defined. There is a function_exists function that determines whether a function with a given name has already been defined. Function calls must use parentheses, with the exception of zero-argument class constructor functions called with the PHP operator new, in which case parentheses are optional.

Until PHP 5.3, support for anonymous functions and closures did not exist in PHP. While create_function() exists since PHP 4.0.1, it is merely a thin wrapper around eval()that allows normal PHP functions to be created during program execution. PHP 5.3 added syntax to define an anonymous function or "closure" which can capture variables from the surrounding scope:

```
function getAdder($x) {
    return function($y) use ($x) {
        return $x + $y;
    };
}

$adder = getAdder(8);
echo $adder(2); // prints "10"
```

In the example above, getAdder() function creates a closure using passed argument $x (the keyword use imports a variable from the lexical context), which takes an additional argument $y, and returns the cr eated closure to the caller. Such a function is a first-class object, meaning that it can be stored in a variable, passed as a parameter to other functions, etc.

Unusually for a dynamically typed language, PHP supports type declarations on function parameters, which are enforced at runtime. This has been supported for classes and interfaces since PHP 5.0, for arrays since PHP 5.1, for "callables" since PHP 5.4, and scalar (integer, float, string and boolean) types since PHP 7.0. PHP 7.0 also has type declarations for function return types, expressed by placing the type name after the list of parameters, preceded by a colon. For example, the getAdder function from the earlier example could be annotated with types like so in PHP 7:

```
function getAdder(int $x): \Closure {
    return function(int $y) use ($x) : int {
        return $x + $y;
    };
}

$adder = getAdder(8);
echo $adder(2);        // prints "10"
echo $adder(null);     // throws an exception because an
incorrect type was passed
$adder = getAdder([]); // would also throw an exception
```

By default, scalar type declarations follow weak typing principles. So, for example, if a parameter's type is int, PHP would allow not only integers, but also convertible numeric strings, floats or Booleans to be passed to that function, and would convert them. However, PHP 7 has a "strict typing" mode which, when used, disallows such conversions for function calls and returns within a file.

Object-oriented programming

Basic object-oriented programming functionality was added in PHP 3 and improved in PHP 4. This allowed for PHP to gain further abstraction, making creative tasks easier for programmers using the language. Object handling was completely rewritten for PHP 5, expanding the feature set and enhancing performance. In previous versions of PHP, objects were handled like value types. The drawback of this method was that code had to make heavy use of PHP's "reference" variables if it wanted to modify an object it was passed rather than creating a copy of it. In the new approach, objects are referenced by handle, and not by value.

PHP 5 introduced private and protected member variables and methods, along with abstract classes, final classes, abstract methods, and final methods. It also introduced a standard way of declaring constructors and destructors, similar to that of other object-oriented languages such as C++, and a standard exception handling model. Furthermore, PHP 5 added interfaces and allowed for multiple interfaces to be implemented. There are special interfaces that allow objects to interact with the runtime system. Objects implementing Array Access can be used with array syntax and objects implementing Iterator or Iterator Aggregate can be used with the for each language construct. There is no virtual table feature in the engine, so static variables are bound with a name instead of a reference at compile time.

If the developer creates a copy of an object using the reserved word clone, the Zend engine will check whether a __clone() method has been defined. If not, it will call a default __clone() which will copy the object's properties. If a __clone() method is defined, then it will be responsible for setting the necessary properties in the created object. For convenience, the engine will supply a function that imports the properties

of the source object, so the programmer can start with a by-value replica of the source object and only override properties that need to be changed.

The following is a basic example of object-oriented programming in PHP:

```
class Person
{
    public $firstName;
    public $lastName;

    public function __construct($firstName, $lastName =
'') { // optional second argument
        $this->firstName = $firstName;
        $this->lastName  = $lastName;
    }

    public function greet() {
        return 'Hello, my name is ' . $this->firstName .
               (($this->lastName != '') ? (' ' . $this-
>lastName) : '') . '.';
    }

    public static function staticGreet($firstName,
$lastName) {
        return 'Hello, my name is ' . $firstName . ' ' .
$lastName . '.';
    }
}

$he    = new Person('John', 'Smith');
$she   = new Person('Sally', 'Davis');
$other = new Person('iAmine');

echo $he->greet(); // prints "Hello, my name is John
Smith."
echo '<br />';

echo $she->greet(); // prints "Hello, my name is Sally
Davis."
echo '<br />';

echo $other->greet(); // prints "Hello, my name is
iAmine."
echo '<br />';

echo Person::staticGreet('Jane', 'Doe'); // prints "Hello,
my name is Jane Doe."
```

The visibility of PHP properties and methods is defined using the keywords public, private, and protected. The default is public, if only var is used; var is a synonym for public. Items declared public can be accessed everywhere. protected limits access to inherited classes (and to the class that defines the item). private limits visibility only to the class that defines the item. Objects of the

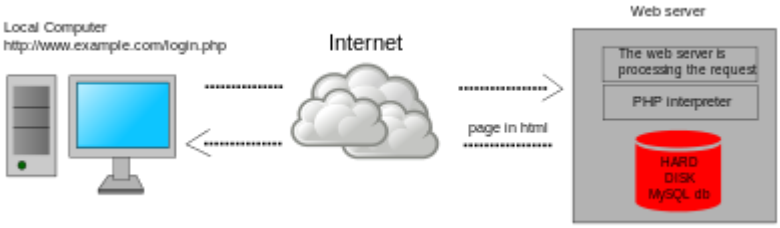
same type have access to each other's private and protected members even though they are not the same instance. PHP's member visibility features have sometimes been described as "highly useful." However, they have also sometimes been described as "at best irrelevant and at worst positively harmful."

Implementations:
The original, only complete and most widely used PHP implementation is powered by the Zend Engine and known simply as PHP. To disambiguate it from other implementations, it is sometimes unofficially referred to as "Zend PHP". The Zend Engine compiles PHP source code on-the-fly into an internal format that it can execute, thus it works as an interpreter. It is also the "reference implementation" of PHP, as PHP has no formal specification, and so the semantics of Zend PHP define the semantics of PHP itself. Due to the complex and nuanced semantics of PHP, defined by how Zend works, it is difficult for competing implementations to offer complete compatibility.

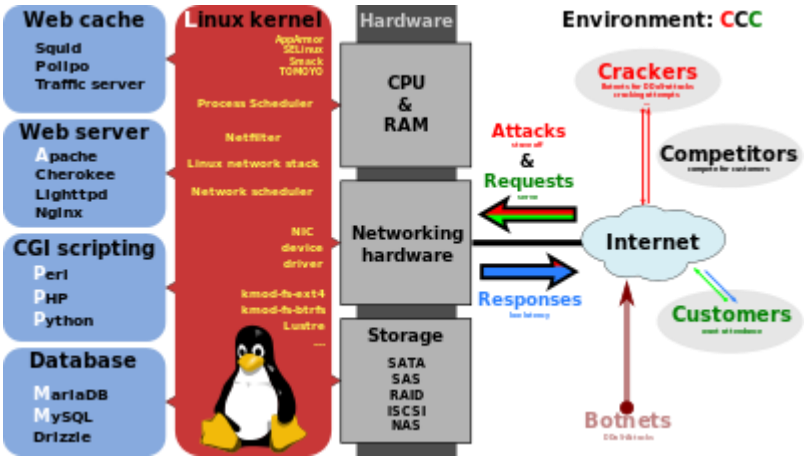
PHP's single-request-per-script-execution model, and the fact that the Zend Engine is an interpreter, leads to inefficiency; as a result, various products have been developed to help improve PHP performance. In order to speed up execution time and not have to compile the PHP source code every time the web page is accessed, PHP scripts can also be deployed in the PHP engine's internal format by using an opcode cache, which works by caching the compiled form of a PHP script (opcodes) in shared memory to avoid the overhead of parsing and compiling the code every time the script runs. An opcode cache, Zend Opcache, is built into PHP since version 5.5. Another example of a widely used opcode cache is the Alternative PHP Cache (APC), which is available as a PECL extension.

While Zend PHP is still the most popular implementation, several other implementations have been developed. Some of these are compilers or support JIT compilation, and hence offer performance benefits over Zend PHP at the expense of lacking full PHP compatibility

Use
PHP is a general-purpose scripting language that is especially suited to server-side web development, in which case PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content or dynamic images used on websites or else where.It can also be used for command-line scripting and client-side graphical user interface (GUI) applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems (RDBMS). Most web hosting providers support PHP for use by their clients. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.



Dynamic web page: example of server-side scripting (PHP and MySQL).



PHP acts primarily as a filter, taking input from a file or stream containing text and/or PHP instructions and outputting another stream of data. Most commonly the output will be HTML, although it could be JSON, XML or binary data such as image or audio formats. Since PHP 4, the PHP parser compiles input to produce bytecode for processing by the Zend Engine, giving improved performance over its interpreter predecessor.

Originally designed to create dynamic web pages, PHP now focuses mainly on server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's ASP.NET, Sun Microsystems' Java Server Pages, and mod_perl. PHP has also attracted the development of many software frameworks that provide building blocks and a design structure to promote rapid application development (RAD). Some of these include PRADO, CakePHP, Symfony, CodeIgniter, Laravel, Yii Framework, Phalcon and Zend Framework, offering features similar to other web frameworks.

The WAMP architecture has become popular in the web industry as a way of deploying web applications. PHP is commonly used as the P in this bundle alongside Linux, Apache and MySQL, although the P may also refer to Python, Perl, or some mix of the three. Similar packages, WAMP and MAMP, are also available for Windows and macOS, with the first letter standing for the respective operating system. Although both PHP and Apache are provided as part of the macOS base install, users of these packages seek a simpler installation mechanism that can be more easily kept up to date.

As of April 2007, over 20 million Internet domains had web services hosted on servers with PHP installed and mod_php was recorded as the most popular Apache HTTP Server module. As of October 2010, PHP was used as the server-side programming language on 75% of all websites whose server-side programming language was known (as of February 2014, the percentage had reached 82%), and PHP was the most-used open source software within enterprises. Web content management

systems written in PHP include MediaWiki, Joomla, eZ Publish, eZ Platform, SilverStripe, WordPress, Drupal, and Moodle. Websites written in PHP, in back-end and/or user-facing portion, include Facebook, Digg, Tumblr, Dailymotion, and Slack.

For specific and more advanced usage scenarios, PHP offers a well defined and documented way for writing custom extensions in C or C++. Besides extending the language itself in form of additional libraries, extensions are providing a way for improving execution speed where it is critical and there is room for improvements by using a true compiled language. PHP also offers well defined ways for embedding itself into other software projects. That way PHP can be easily used as an internal scripting language for another project, also providing tight interfacing with the project's specific internal data structures.

PHP received mixed reviews due to lacking support for multithreading at the core language level, though using threads is made possible by the "pthreads" PECL extension.

HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current version of the HTML standard.

It was published in October 2014 by the World Wide Web Consortium (W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc. HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

Figure 11 HTML 5

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-

platform mobile applications, because it includes features designed with low-powered devices in mind.

Many new syntactic features are included. To natively include and handle multimedia and graphical content, the new <video>, <audio> and <canvas> elements were added, and support for scalable vector graphics (SVG) content and MathML for mathematical formulas. To enrich the semantic content of documents, new page structure elements such as <main>, <section>, <article>, <header>, <footer>, <aside>, <nav> and <figure>, are added. New attributes are introduced, some elements and attributes have been removed, and others such as <a>, <cite> and <menu> have been changed, redefined or standardized.

The APIs and Document Object Model (DOM) are now fundamental parts of the HTML5 specification and HTML5 also better defines the processing for any invalid documents.

FEATURES

Mark-up: HTML5 introduces elements and attributes that reflect typical usage on modern websites. Some of them are semantic replacements for common uses of generic block ( ) and inline () elements, for example <nav> (website navigation block), <footer> (usually referring to bottom of web page or to last lines of HTML code), or <audio>and <video> instead of <object>. Some deprecated elements from HTML 4.01 have been dropped, including purely presentational elements such as <font> and , whose effects have long been superseded by the more capable Cascading Style Sheets. There is also a renewed emphasis on the importance of DOM scripting (e.g., JavaScript) in Web behavior.

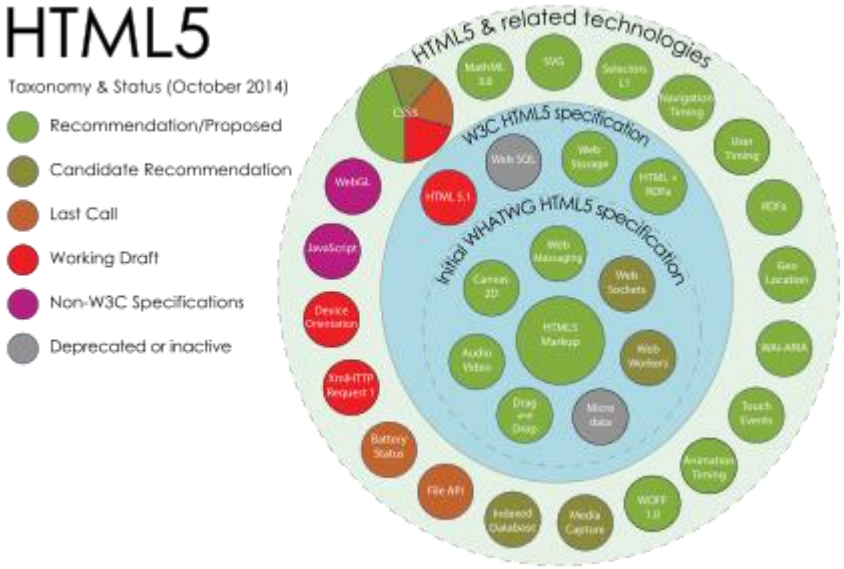
The HTML5 syntax is no longer based on SGML despite the similarity of its markup. It has, however, been designed to be backward compatible with common parsing of older versions of HTML. It comes with a new introductory line that looks like an SGML document type declaration, <!DOCTYPE html>, which triggers the standards-compliant rendering mode.https://en.wikipedia.org/wiki/HTML5 Since 5 January 2009, HTML5 also includes Web Forms 2.0, a previously separate WHATWG specification. Since 5 January 2009, HTML5 also includes Web Forms 2.0, a previously separate WHATWG specification.

HTML5 related APIs

In addition to specifying markup, HTML5 specifies scripting application programming interfaces (APIs) that can be used with JavaScript. Existing document object model (DOM) interfaces are extended and de facto features documented. There are also new APIs, such as:

- Canvas;
- Timed Media Playback;
- Offline;
- Editable content;
- Drag-and-drop
- History
- MIME type and protocol handler registration;
- Microdata;

- Web Messaging;



Web Storage – a key-value pair storage framework that provides behaviour similar to cookies but with larger storage capacity and improved API.

Not all of the above technologies are included in the W3C HTML5 specification, though they are in the WHATWG HTML specification. Some related technologies, which are not part of either the W3C HTML5 or the WHATWG HTML specification, are as follows. The W3C publishes specifications for these separately:

Geolocation

Web SQL Database – a local SQL Database (no longer maintained);

Indexed DB – an indexed hierarchical key-value store (formerly Web Simple DB);

File – an API intended to handle file uploads and file manipulation;

Directories and System – an API intended to satisfy client-side-storage use cases not well served by databases;

File Writer – an API for writing to files from web applications;

Web Audio – a high-level JavaScript API for processing and synthesizing audio in web applications;

Class List.

Web Cryptography

Web RTC

HTML5 cannot provide animation within web pages. Additional JavaScript or CSS3 functionality is necessary for animating HTML elements. Animation is also possible using JavaScript and HTML 4, and within SVG elements through SMIL, although browser support of the latter remains uneven as of 2011.

Error handling

HTML5 is designed so that old browsers can safely ignore new HTML5 constructs. In contrast to HTML 4.01, the HTML5 specification gives detailed rules
for lexing and parsing, with the intent that compliant browsers will produce the same results when parsing incorrect syntax.https://en.wikipedia.org/wiki/HTML5 Although HTML5 now defines a consistent behavior for "tag soup" documents, those documents are not regarded as conforming to the HTML5 standard. Although HTML5 now defines a consistent behavior for "tag soup" documents, those documents are not regarded as conforming to the HTML5 standard.

Popularity

According to a report released on 30 September 2011, 34 of the world's top 100 Web sites were using HTML5 – the adoption led by search engines and social networks. Another report released in August 2013 has shown that 153 of the Fortune 500 U.S. companies implemented HTML5 on their corporate websites.

Since 2014, HTML5 is at least partially supported by most popular layout engines.

Differences from HTML 4.01 and XHTML 1.x

The following is a cursory list of differences and some specific examples.

New parsing rules: oriented towards flexible parsing and compatibility; not based on SGML

Ability to use inline SVG and MathML in text/html

New elements: article, aside, audio, bdi, canvas, command, data, datalist, details, embe d, figcaption, figure, footer, header, keygen, mark, meter, nav, output, progress, rp, rt, ruby, section, source, summary, time, track, video, wbr

New types of form controls: dates and
times, email, url, search, number, range, tel, color

New attributes: charset (on meta), a sync (on script)

Global attributes (that can be applied for every element): id, tab index, hidden, data-* (custom data attributes)

Deprecated elements will be dropped
altogether: acronym, applet, basefont, big, center, dir, font, frame, frameset, isindex, n oframes, strike, tt

dev.w3.org provides the latest Editors Draft of "HTML5 differences from HTML 4", which provides a complete outline of additions, removals and changes between HTML5 and HTML 4.

CSS3

CSS3 offers a huge variety of new ways to create an impact with your designs, with quite a few important changes. This first tutorial will give you a very basic introduction to the new possibilities created by the standard.

Figure 12 CSS3

Modules in CSS3

The development of CSS3 is going to be split up into 'modules'. The old specification was simply too large and complex to be updated as one, so it has been broken down into smaller pieces – with new ones also added. Some of these modules include:

- The Box Model
- Lists Module
- Hyperlink Presentation
- Speech Module
- Backgrounds and Borders
- Text Effects
- Multi-Column Layout

Several of the modules have now been completed, including SVG (Scalable Vector Graphics), Media Queries and Namespaces. The others are still being worked upon.

It is incredibly difficult to give a projected date when web browsers will adopt the new features of CSS3 – some new builds of Safari have already started to.

New features will be implemented gradually in different browsers, and it could still be a year or two before every module is widely adopted.

Hopefully, in a mainly positive way. CSS3 will obviously be completely backwards compatible, so it won't be necessary to change existing designs to ensure they work – web browsers will always continue to support CSS2.

The main impact will be the ability to use new selectors and properties which are available. These will allow you to both achieve new design features (animation or gradients for instance), and achieve current design features in a much easier way (e.g. using columns).

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This

separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

Syntax
CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.

A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

Selector
In CSS, selectors declare which part of the markup a style applies to by matching tags and attributes in the markup itself.

Selectors may apply to:

- all elements of a specific type, e.g. the second-level headers h2
- elements specified by attribute, in particular:
- id: an identifier unique within the document
- class: an identifier that can annotate multiple elements in a document
- Elements depending on how they are placed relative to others in the document tree.

Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element.

Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is :hover, which identifies content only when the user "points to" the visible element,

usually by holding the mouse cursor over it. It is appended to a selector as in a:hover or #elementid:hover. A pseudo-class classifies document elements, such as :link or :visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as ::first-line or ::first-letter.

Selectors may be combined in many ways to achieve great specificity and flexibility.[6] Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .myClass {color: red;} applies to all elements of class myClass that are inside div elements, whereas .myClass div {color: red;} applies to all div elements that are in elements of class myClass.

Use

Before CSS, nearly all presentational attributes of HTML documents were contained within the HTML markup. All font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS lets authors move much of that information to another file, the style sheet, resulting in considerably simpler HTML.

For example, headings (h1 elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is presentational.

Before CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to repeat HTML presentational markup for each occurrence of that heading type. This made documents more complex, larger, and more error-prone and difficult to maintain. CSS allows the separation of presentation from structure. CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics, and can do so independently for on-screen and printed views. CSS also defines non-visual styles, such as reading speed and emphasis for aural text readers. The W3C has now deprecated the use of all presentational HTML markup.

For example, under pre-CSS HTML, a heading element defined with red text would be written as:

```
<h1><font color="red"> Chapter 1. </font></h1>
```

Using CSS, the same element can be coded using style properties instead of HTML presentational attributes:

```
<h1 style="color: red;"> Chapter 1. </h1>
```

An "external" CSS file, as described below, can be associated with an HTML document using the following syntax:

```
<link href="path/to/file.css" rel="stylesheet"
type="text/css">
```

An internal CSS code can be typed in the head section of the code. The coding is started with the style tag. For example,

```
<style>
h1 {color: red;}
</style>
```

Limitations

Some noted limitations of the current capabilities of CSS include:

Selectors are unable to ascend

CSS currently offers no way to select a parent or ancestor of an element that satisfies certain criteria. CSS Selectors Level 4, which is still in Working Draft status, proposes such a selector, but only as part of the "complete" selector profile, not the "fast" profile used in dynamic CSS styling. A more advanced selector scheme (such as XPath) would enable more sophisticated style sheets. The major reasons for the CSS Working Group previously rejecting proposals for parent selectors are related to browser performance and incremental rendering issues.

Cannot explicitly declare new scope independently of position

Scoping rules for properties such as z-index look for the closest parent element with a position: absolute or position: relative attribute. This odd coupling has undesired effects. For example, it is impossible to avoid declaring a new scope when one is forced to adjust an element's position, preventing one from using the desired scope of a parent element.

Pseudo-class dynamic behaviour not controllable

CSS implements pseudo-classes that allow a degree of user feedback by conditional application of alternate styles. One CSS pseudo-class, ":hover", is dynamic (equivalent of JavaScript "onmouseover") and has potential for abuse (e.g., implementing cursor-proximity popups),https://en.wikipedia.org/wiki/Cascading_Style_Sheets but CSS has no ability for a client to disable it (no "disable"-like property) or limit its effects (no "no change"-like values for each property).

Cannot name rules

There is no way to name a CSS rule, which would allow (for example) client-side scripts to refer to the rule even if its selector changes.

Cannot include styles from a rule into another rule

CSS styles often must be duplicated in several rules to achieve a desired effect, causing additional maintenance and requiring more thorough testing. Some new CSS features were proposed to solve this, but (as of February, 2016) are not yet implemented anywhere.

Cannot target specific text without altering markup

Besides the :first-letter pseudo-element, one cannot target specific ranges of text without needing to utilize place-holder elements.

Resolved Limitations

Vertical control limitations

Though horizontal placement of elements was always generally easy to control, vertical placement was frequently unintuitive, convoluted, or outright impossible. Simple tasks, such as centering an element vertically or placing a footer no higher than bottom of the viewport required either complicated and unintuitive style rules, or simple but widely unsupported rules. The Flexible Box Module improved the situation considerably and vertical control is much more straightforward and supported in all of the modern

browsers. Older browsers still have those issues, but most of those (mainly Internet Explorer 9 and below) are no longer supported by their vendors.

Absence of expressions

There was no standard ability to specify property values as simple expressions (such as margin-left: 10% – 3em + 4px;). This would be useful in a variety of cases, such as calculating the size of columns subject to a constraint on the sum of all columns. Internet Explorer versions 5 to 7 support a proprietary expression() statement,with similar functionality. This proprietary expression() statement is no longer supported from Internet Explorer 8 onwards, except in compatibility modes. This decision was taken for "standards compliance, browser performance, and security reasons". However, a candidate recommendation with a calc() value to address this limitation has been published by the CSS WG and has since been supported in all of the modern browsers.

Lack of column declaration

Although possible in current CSS 3 (using the column-count module), layouts with multiple columns can be complex to implement in CSS 2.1. With CSS 2.1, the process is often done using floating elements, which are often rendered differently by different browsers, different computer screen shapes, and different screen ratios set on standard monitors. All of the modern browsers support this CSS 3 feature in one form or another.

Advantages

Separation of content from presentation

CSS facilitates publication of content in multiple presentation formats based on nominal parameters. Nominal parameters include explicit user preferences, different web browsers, the type of device being used to view the content (a desktop computer or mobile Internet device), the geographic location of the user and many other variables.

Site-wide consistency

When CSS is used effectively, in terms of inheritance and "cascading", a global style sheet can be used to affect and style elements site-wide. If the situation arises that the styling of the elements should be changed or adjusted, these changes can be made by editing rules in the global style sheet. Before CSS, this sort of maintenance was more difficult, expensive and time-consuming.

Bandwidth

A stylesheet, internal or external, specifies the style once for a range of HTML elements selected by class, type or relationship to others. This is much more efficient than repeating style information inline for each occurrence of the element. An external stylesheet is usually stored in the browser cache, and can therefore be used on multiple pages without being reloaded, further reducing data transfer over a network.

Page reformatting

With a simple change of one line, a different style sheet can be used for the same page. This has advantages for accessibility, as well as providing the ability to tailor a page or site to different target devices. Furthermore, devices not able to understand the styling still display the content.

Accessibility

Without CSS, web designers must typically lay out their pages with techniques such as HTML tables that hinder accessibility for vision-impaired users (see Tableless web design#Accessibility).

CSS Framework

CSS frameworks are pre-prepared libraries that are meant to allow for easier, more standards-compliant styling of web pages using the Cascading Style Sheets language. CSS frameworks include Foundation, Blueprint, Bootstrap, Cascade Framework and Materialize. Like programming and scripting language libraries, CSS frameworks are usually incorporated as external .css sheets referenced in the HTML <head>. They provide a number of ready-made options for designing and laying out the web page. Although many of these frameworks have been published, some authors use them mostly for rapid prototyping, or for learning from, and prefer to 'handcraft' CSS that is appropriate to each published site without the design, maintenance and download overhead of having many unused features in the site's styling.

JavaScript

JavaScript is a very powerful client-side scripting language. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage more lively and interactive, with the help of JavaScript. JavaScript is also being used widely in game development and Mobile application development.

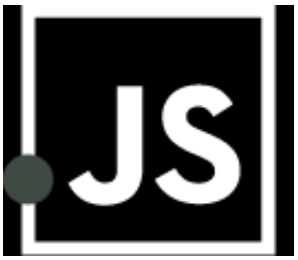


Figure 13 JavaScript JS

JavaScript is also used in environments that are not Web-based, such as PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines (VMs) and platforms built upon them have also increased the popularity of JavaScript for server-side Web applications. On the client side, developers have traditionally implemented JavaScript as an interpreted language, but more recent browsers perform just-in-time compilation. Programmers also use JavaScript in video-game development, in crafting desktop and mobile applications, and in server-side network programming with run-time environments such as Node.js.

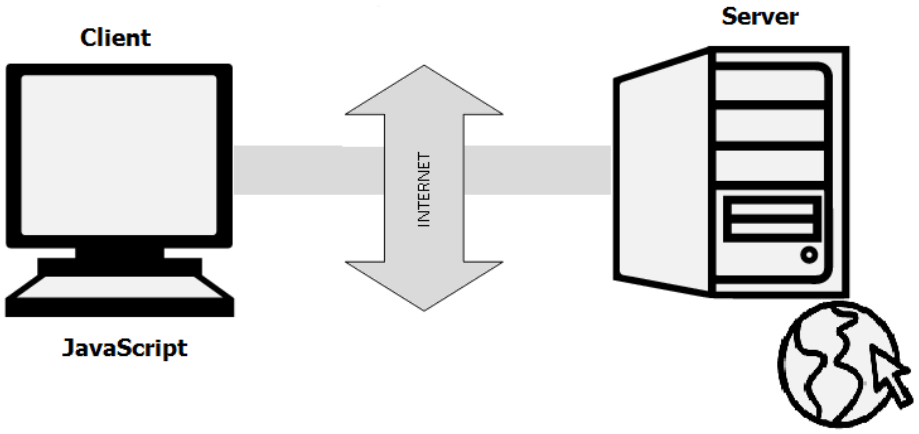


Figure 14 Client Server Relation

Being a scripting language, JavaScript cannot run on its own. In fact, the browser is responsible for running JavaScript code. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it is up to the browser to execute it. The main advantage of JavaScript is that all modern web browsers support JavaScript. So, you do not have to worry whether your site visitor uses Internet Explorer, Google Chrome, Firefox or any other browser. JavaScript will be supported. Also, JavaScript runs on any operating system including Windows, Linux or Mac. Thus, JavaScript overcomes the main disadvantages of VBScript which is limited to just IE and Windows.

Tools You Need In JavaScript: To start with, you need a text editor to write your code and a browser to display the web pages you develop. You can use text editor of your choice including Notepad++, Komodo Edit or any other text editor you are comfortable with. You can use any web browser including Internet Explorer, Google Chrome, Firefox etc.

Features

The following features are common to all conforming ECMAScript implementations, unless explicitly specified otherwise.

Universal support:

All modern Web browsers support JavaScript with built-in interpreters.

Imperative and structured:

JavaScript supports much of the structured programming syntax from C (e.g., if statements, while loops, switch statements, do while loops, etc.). One partial exception is scoping: JavaScript originally had only function scoping with var. ECMAScript 2015 added a let keyword for block scoping, meaning JavaScript now has both function and block scoping. Like C, JavaScript makes a distinction between expressions and statements. One syntactic difference from C is automatic semicolon insertion, which allows the semicolons that would normally terminate statements to be omitted.

Dynamic

Typing: As with most scripting languages, JavaScript is dynamically typed; a type is associated with each value, rather than just with each expression. For example,

a variable that is at one time bound to a number may later be re-bound to a string.https://en.wikipedia.org/wiki/JavaScript JavaScript supports various ways to test the type of an object, including duck typing.

Run-time evaluation

JavaScript includes an eval function that can execute statements provided as strings at run-time.

Prototype-based (Object-oriented)

JavaScript is almost entirely object-based. In JavaScript, an object is an associative array, augmented with a prototype (see below); each string key provides the name for an object property, and there are two syntactical ways to specify such a name: dot notation (obj.x = 10) and bracket notation (obj['x'] = 10). A property may be added, rebound, or deleted at run-time. Most properties of an object (and any property that belongs to an object's prototype inheritance chain) can be enumerated using a for...inloop.

JavaScript has a small number of built-in objects, including Function and Date.

Prototypes

JavaScript uses prototypes where many other object-oriented languages use classes for inheritance. It is possible to simulate many class-based features with prototypes in JavaScript.

Functions as object constructors

Functions double as object constructors, along with their typical role. Prefixing a function call with new will create an instance of a prototype, inheriting properties and methods from the constructor (including properties from the Object prototype). ECMAScript 5 offers the Object.create method, allowing explicit creation of an instance without automatically inheriting from the Object prototype (older environments can assign the prototype to null). The constructor's prototype property determines the object used for the new object's internal prototype. New methods can be added by modifying the prototype of the function used as a constructor. JavaScript's built-in constructors, such as Array or Object, also have prototypes that can be modified. While it is possible to modify the Object prototype, it is generally considered bad practice because most objects in JavaScript will inherit methods and properties from the Object prototype, and they may not expect the prototype to be modified.

Functions as methods

Unlike many object-oriented languages, there is no distinction between a function definition and a method definition. Rather, the distinction occurs during function calling; when a function is called as a method of an object, the functions' local this keyword is bound to that object for that invocation.

Functional

A function is first-class; a function is considered to be an object. As such, a function may have properties and methods, such as .call() and .bind(). A nested function is a function defined within another function. It is created each time the outer function is invoked. In addition, each nested function forms a lexical closure: The lexical scope of the outer function (including any constant, local variable, or argument value) becomes

part of the internal state of each inner function object, even after execution of the outer function concludes. JavaScript also supports anonymous functions.

Delegative

JavaScript supports implicit and explicit delegation.

Functions as roles (Traits and Mixins)

JavaScript natively supports various function-based implementations of Role patterns like Traits and Mixins. Such a function defines additional behavior by at least one method bound to the this keyword within its function body. A Role then has to be delegated explicitly via call or apply to objects that need to feature additional behavior that is not shared via the prototype chain.

Object composition and inheritance

Whereas explicit function-based delegation does cover composition in JavaScript, implicit delegation already happens every time the prototype chain is walked in order to, e.g., find a method that might be related to but is not directly owned by an object. Once the method is found it gets called within this object's context. Thus inheritance in JavaScript is covered by a delegation automatism that is bound to the prototype property of constructor functions.

Miscellaneous

Run-time environment: JavaScript typically relies on a run-time environment (e.g., a Web browser) to provide objects and methods by which scripts can interact with the environment (e.g., a webpage DOM). It also relies on the run-time environment to provide the ability to include/import scripts (e.g., HTML <script> elements). This is not a language feature per se, but it is common in most JavaScript implementations.

JavaScript processes messages from a queue one at a time. Upon loading a new message, JavaScript calls a function associated with that message, which creates a call stack frame (the function's arguments and local variables). The call stack shrinks and grows based on the function's needs. Upon function completion, when the stack is empty, JavaScript proceeds to the next message in the queue. This is called the event loop, described as "run to completion" because each message is fully processed before the next message is considered. However, the language's concurrency model describes the event loop as non-blocking: program input/output is performed using events and callback functions. This means, for instance, that JavaScript can process a mouse click while waiting for a database query to return information.

Variadic functions

An indefinite number of parameters can be passed to a function. The function can access them through formal parameters and also through the local arguments object. Variadic functions can also be created by using the bind method.

Array and object literals

Like many scripting languages, arrays and objects (associative arrays in other languages) can each be created with a succinct shortcut syntax. In fact, these literals form the basis of the JSON data format.

Regular expressions

JavaScript also supports regular expressions in a manner similar to Perl, which provide a concise and powerful syntax for text manipulation that is more sophisticated than the built-in string functions.

Vendor-specific extensions

JavaScript is officially managed by Mozilla Foundation, and new language features are added periodically. However, only some JavaScript engines support these new features:

- property getter and setter functions (supported by WebKit, Gecko, Opera, ActionScript, and Rhino)
- conditional catch clauses
- iterator protocol (adopted from Python)
- shallow generators-coroutines (adopted from Python)
- array comprehensions and generator expressions (adopted from Python)
- proper block scope via the let keyword
- array and object destructuring (limited form of pattern matching)
- concise function expressions (function(args) expr)
- ECMAScript for XML (E4X), an extension that adds native XML support to ECMAScript (unsupported in Firefox since version 21)

Syntax:

Simple examples

Variables in JavaScript can be defined using the var keyword:

```
var x; // defines the variable x and assigns to it the
special value "undefined" (not to be confused with an
undefined value)
var y = 2; // defines the variable y and assigns to it the
value 2
```

Note the comments in the example above, both of which were preceded with two forward slashes.

There is no built-in I/O functionality in JavaScript; the run-time environment provides that. The ECMAScript specification in edition 5.1 mentions:

Indeed, there are no provisions in this specification for input of external data or output of computed results.

However, most runtime environments have a console object that can be used to print output. Here is a minimalist Hello World program in JavaScript:

```
console.log("Hello World!");
```

A simple recursive function:

```
function factorial(n) {
    if (n === 0 || n === 1) {
        return 1;  // 0! = 1! = 1
    }
    return n * factorial(n - 1);
}
factorial(3); // returns 6
```

An anonymous function (or lambda):

```
function counter() {
    var count = 0;
    return function() {
        return ++count;
    };
}
var closure = counter();
closure(); // returns 1
closure(); // returns 2
closure(); // returns 3
```

```
function sum() {
// This example shows that in JavaScript, function
closures captures their non-local variables by reference.
//Variadic function demonstration (arguments is a
special variable):
    var x = 0;
    for (var i = 0; i < arguments.length; ++i) {
        x += arguments[i];
    }
    return x;
}
sum(1, 2); // returns 3
sum(1, 2, 3); // returns 6
```

Immediately-invoked function expressions are often used to create modules, as before ECMAScript 2015 there was not built-in construct in the language. Modules allow gathering properties and methods in a namespace and making some of them private:

```
var counter = (function () {
    var i = 0; // private property
    return {   // public methods
        get: function () {
            alert(i);
        },
        set: function (value) {
            i = value;
        },
        increment: function () {
            alert(++i);
        }
```

```
    };
})(); // module
counter.get();       // shows 0
counter.set(6);
counter.increment(); // shows 7
counter.increment(); // shows 8
```

Twitter Bootstrap3

Bootstrap is a powerful front-end framework for faster and easier web development. It includes HTML and CSS based design templates for common user interface components like Typography, Forms, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel and many other as well as optional JavaScript extensions.

Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden. According to Twitter developer Mark Otto:

"A super small group of developers and I got together to design and build a new internal tool and saw an opportunity to do something more. Through that process, we saw ourselves build something much more substantial than another internal tool. Months later, we ended up with an early version of Bootstrap as a way to document and share common design patterns and assets within the company."

Figure 15 Bootstrap 3

After a few months of development by a small group, many developers at Twitter began to contribute to the project as a part of Hack Week, a hackathon-style week for the Twitter development team. It was renamed from Twitter Blueprint to Bootstrap, and released as an open source project on August 19, 2011. It has continued to be maintained by Mark Otto, Jacob Thornton, and a small group of core developers, as well as a large community of contributors.

On January 31, 2012, Bootstrap 2 was released, which added a twelve-column responsive grid layout system, inbuilt support for Glyph icons, several new components, as well as changes to many of the existing components.

On August 19, 2013, Bootstrap 3 was released, which redesigned components to use flat design, and a mobile first approach.

On October 29, 2014, Mark Otto announced that Bootstrap 4 was in development. The first alpha version of Bootstrap 4 was released on August 19, 2015.

Structure and Functions

Bootstrap is modular and consists of a series of Less stylesheets that implement the various components of the toolkit. These stylesheets are generally compiled into a bundle and included in web pages, but individual components can be included or removed. Bootstrap provides a number of configuration variables that control things such as color and padding of various components.

Since Bootstrap 2, the Bootstrap documentation has included a customization wizard which generates a customized version of Bootstrap based on the requested components and various settings.

As of Bootstrap 4, Sass is used instead of Less for the stylesheets.

Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code.

Grid system and responsive design comes standard with an 1170 pixel wide grid layout. Alternatively, the developer can use a variable-width layout. For both cases, the toolkit has four variations to make use of different resolutions and types of devices: mobile phones, portrait and landscape, tablets and PCs with low and high resolution. Each variation adjusts the width of the columns.

Stylesheets

Bootstrap provides a set of stylesheets that provide basic style definitions for all key HTML components. These provide a uniform, modern appearance for formatting text, tables and form elements.

Re-usable components

In addition to the regular HTML elements, Bootstrap contains other commonly used interface elements. The components are implemented as CSS classes, which must be applied to certain HTML elements in a page.

JavaScript components

Bootstrap comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields. In version 2.0, the following JavaScript plugins are supported: Modal, Dropdown, Scroll spy, Tab, Tooltip, Popover, Alert, Button, Collapse, Carousel and Type ahead.

Bootstrap also gives you ability to create responsive layout with much less efforts.

Advantages of Bootstrap

The biggest advantage of using Bootstrap is that it comes with free set of tools for creating flexible and responsive web layouts as well as common interface components.

Additionally, using the Bootstrap data APIs you can create advanced interface components like Scroll spy and Type heads without writing a single line of JavaScript.

Here are some more advantages, why one should opt for Bootstrap:

Save lots of time

You can save lots of time and efforts using the Bootstrap predefined design templates and classes and concentrate on other development work.

Responsive features

Using Bootstrap you can easily create responsive designs. Bootstrap responsive features make your web pages to appear more appropriately on different devices and screen resolutions without any change in mark-up.

Consistent design

All Bootstrap components share the same design templates and styles through a central library, so that the designs and layouts of your web pages are consistent throughout your development.

Easy to use

Bootstrap is very easy to use. Anybody with the basic working knowledge of HTML and CSS can start development with Bootstrap.

Compatible with browsers

Bootstrap is created with modern browsers in mind and it is compatible with all modern browsers such as Mozilla Firefox, Google Chrome, Safari, Internet Explorer, and Opera.

Open Source

The best part is, it is completely free to download and use.

LESS

Less (sometimes stylized as LESS) is a dynamic style sheet language that can be compiled into Cascading Style Sheets (CSS) and run on the client side or server side. Designed by Alexis Sellier, Less is influenced by Sass and has influenced the newer "SCSS" syntax of Sass, which adapted its CSS-like block formatting syntax. Less is open source.

Figure 16 LESS (Less CSS)

Its first version was written in Ruby; however, in the later versions, use of Ruby has been deprecated and replaced by JavaScript. The indented syntax of Less is a nested metalanguage, as valid CSS is valid Less code with the same semantics. Less provides the following mechanisms: variables, nesting, mixes, operators and functions; the main difference between Less and other CSS precompiles being that Less allows real-time compilation via less.js by the browser.

Variables

These are pretty self-explanatory:

```
@nice-blue: #5B83AD;
@light-blue: @nice-blue + #111;
#header {
  color: @light-blue;
}
Outputs
#header {
  color: #6c94be;
}
```

Note that variables are actually "constants" in that they can only be defined once.

Mixins

Mixins are a way of including ("mixing in") a bunch of properties from one rule-set into another rule-set. So say we have the following class:

```
.bordered {
  border-top: dotted 1px black;
  border-bottom: solid 2px black;
}
```

And we want to use these properties inside other rule-sets. Well, we just have to drop in the name of the class where we want the properties, like so:

```
#menu a {
  color: #111;
  .bordered;
}
.post a {
  color: red;
  .bordered;
}
```

The properties of the .bordered class will now appear in both #menu a and .post a. (Note that you can also use #ids as mixins.)

Nested Rules

Less gives you the ability to use nesting instead of, or in combination with cascading. Let's say we have the following CSS:

```
#header {
  color: black;
}
#header .navigation {
  font-size: 12px;
```

```
}
#header .logo {
  width: 300px;
}
```

In Less, we can also write it this way:

```
#header {
  color: black;
  .navigation {
    font-size: 12px;
  }
  .logo {
    width: 300px;
  }
}
```

The resulting code is more concise, and mimics the structure of your HTML.

You can also bundle pseudo-selectors with your mixins using this method. Here's the classic clearfix hack, rewritten as a mixin (& represents the current selector parent):

```
.clearfix {
  display: block;
  zoom: 1;
  &:after {
    content: " ";
    display: block;
    font-size: 0;
    height: 0;
    clear: both;
    visibility: hidden;
  }
}
```

Nested Directives and Bubbling

Directives such as media or keyframe can be nested in the same way as selectors. Directive is placed on top and relative order against other elements inside the same ruleset remains unchanged. This is called bubbling.

Conditional directives e.g. @Media, @supports and @document have also selectors copied into their bodies:

```
.screen-color {
  @media screen {
    color: green;
    @media (min-width: 768px) {
      color: red;
    }
  }
  @media tv {
    color: black;
  }
}
```

outputs:

```
@media screen {
  .screen-color {
    color: green;
  }
}
@media screen and (min-width: 768px) {
  .screen-color {
    color: red;
  }
}
@media tv {
  .screen-color {
    color: black;
  }
}
```

Remaining non-conditional directives, for example font-face or keyframes, are bubbled up too. Their bodies do not change:

```
#a {
  color: blue;
  @font-face {
    src: made-up-url;
  }
  padding: 2 2 2 2;
}

outputs:

#a {
  color: blue;
}
@font-face {
  src: made-up-url;
}
#a {
  padding: 2 2 2 2;
}
```

Operations

Arithmetical operations +, -, *, / can operate on any number, color or variable. If it is possible, mathematical operations take units into account and convert numbers before adding, subtracting or comparing them. The result has leftmost explicitly stated unit type. If the conversion is impossible or not meaningful, units are ignored. Example of impossible conversion: px to cm or rad to %.

```
// numbers are converted into the same units
@conversion-1: 5cm + 10mm; // result is 6cm
@conversion-2: 2 - 3cm - 5mm; // result is -1.5cm
```

```
// conversion is impossible
@incompatible-units: 2 + 5px - 3cm; // result is 4px

// example with variables
@base: 5%;
@filler: @base * 2; // result is 10%
@other: @base + @filler; // result is 15%
```

Multiplication and division do not convert numbers. It would not be meaningful in most cases - a length multiplied by a length gives an area and css does not support specifying areas. Less will operate on numbers as they are and assign explicitly stated unit type to the result.

```
@base: 2cm * 3mm; // result is 6cm
```

Colors are split into their red, green, blue and alpha dimensions. The operation is applied to each color dimension separately. E.g., if the user added two colors, then the green dimension of the result is equal to sum of green dimensions of input colors. If the user multiplied a color by a number, each color dimension will get multiplied.

Note: arithmetic operation on alpha is not defined, because math operation on colors do not have standard agreed upon meaning. Do not rely on current implemention as it may change in later versions.

An operation on colors always produces valid color. If some color dimension of the result ends up being bigger than ffor smaller than 00, the dimension is rounded to either ff or 00. If alpha ends up being bigger than 1.0 or smaller than 0.0, the alpha is rounded to either 1.0 or 0.0.

```
@color: #224488 / 2; //results in #112244
background-color: #112244 + #111; // result is #223355
```

Escaping

Escaping allows you to use any arbitrary string as property or variable value. Anything inside ~"anything" or ~'anything' is used as is with no changes except interpolation.

```
.weird-element {
  content: ~"^//* some horrible but needed css hack";
}
results in:
.weird-element {
  content: ^//* some horrible but needed css hack;
}
```

Functions

Less provides a variety of functions which transform colors, manipulate strings and do maths. They are documented fully in the function reference.

Using them is pretty straightforward. The following example uses percentage to convert 0.5 to 50%, increases the saturation of a base color by 5% and then sets the background color to one that is lightened by 25% and spun by 8 degrees:

```
@base: #f04615;
@width: 0.5;
.class {
  width: percentage(@width); // returns `50%`
  color: saturate(@base, 5%);
  background-color: spin(lighten(@base, 25%), 8);
}
```

MySQL

MySQL is a fast, easy-to-use RDBMS (Relation Database Management System) being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.

Figure 17 My SQL

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000https://en.wikipedia.org/wiki/MySQL (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license.

Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona.

MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is

very, very good". It has also been tested to be a "fast, stable and true multi-user, multi-threaded sql database server". MySQL is becoming so popular because of many good reasons:

- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
- MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

Features

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL 5.6

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM
- Triggers
- Cursors
- Updatable views
- Online DDL when using the InnoDB Storage Engine.
- Information schema
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines
- SSL support
- Query caching
- Sub-SELECTs (i.e. nested SELECTs)

- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master. Multi-master replication is provided in MySQL Cluster, and multi-master support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching
- Embedded database library
- Unicode support
- Partitioned tables with pruning of partitions in optimizer
- Shared-nothing clustering through MySQL Cluster
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

The developers release minor updates of the MySQL Server approximately every two months. The sources can be obtained from MySQL's website or from MySQL's GitHub repository, both under the GPL license.

User Interfaces

Graphical user interfaces

A graphical user interface (GUI) is a type of interface that allows users to interact with electronic devices or programs through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. GUIs are easier to learn than command-line interfaces (CLIs), which require commands to be typed on the keyboard.

Third-party proprietary and free graphical administration applications (or "front ends") are available that integrate with MySQL and enable users to work with database structure and data visually. Some well-known front ends are:

Figure 18 MySQL Workbench running on macOS

MySQL Workbench

MySQL Workbench is the official integrated environment for MySQL. It was developed by MySQL AB, and enables users to graphically administer MySQL databases and

visually design database structures. MySQL Workbench replaces the previous package of software, MySQL GUI Tools. Similar to other third-party packages, but still considered the authoritative MySQL front end, MySQL Workbench lets users manage database design & modelling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).

MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition.

Adminer

Adminer (formerly known as phpMinAdmin) is a free MySQL front end for managing content in MySQL databases (since version 2, it also works on PostgreSQL, MS SQL, SQLite and Oracle SQL databases). Adminer is distributed under the Apache license (or GPL v2) in the form of a single PHP file (around 300 KiB in size), and is capable of managing multiple databases, with many CSS skins available. Its author is Jakub Vrána who started to develop this tool as a light-weight alternative to phpMyAdmin, in July 2007.

Database Workbench

Database Workbench is a software application for development and administration of multiple relational databases using SQL, with interoperationality between different database systems, developed by Upscene Productions.

Because Databases Workbench supports multiple database systems, it can provide software developers with the same interface and development environment for these otherwise different database systems and also includes cross database tools.

Database Workbench supports the following relational databases: Oracle Database, Microsoft SQL Server, SQL Anywhere, Firebird, NexusDB, InterBase, MySQL and MariaDB. Database Workbench 5 runs on 32-bit or 64 bit Windows platforms. Under Linux, FreeBSD or macOS Database Workbench can operate using Wine.

LibreOffice Base

LibreOffice Base allows the creation and management of databases, preparation of forms and reports that provide end users easy access to data. Like Microsoft Access, it can be used as a front-end for various database systems, including Access databases (JET), ODBC data sources, and MySQL or PostgreSQL

OpenOffice.org

OpenOffice.org Base is freely available and can manage MySQL databases if the entire suite is installed.

PhpMyAdmin

phpMyAdmin is a free and open source tool written in PHP intended to handle the administration of MySQL with the use of a web browser. It can perform various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions. The software, which is available in 78 languages, is maintained by The phpMyAdmin Project.

It can import data from CSV and SQL, and transform stored data into any format using a set of predefined functions, like displaying BLOB-data as images or download-links.

Command-line interfaces

A command-line interface is a means of interacting with a computer program where the user issues commands to the program by typing in successive lines of text (command lines). MySQL ships with many command line tools, from which the main interface is the mysql client.

MySQL Utilities is a set of utilities designed to perform common maintenance and administrative tasks. Originally included as part of the MySQL Workbench, the utilities are a stand-alone download available from Oracle.

Percona Toolkit is a cross-platform toolkit for MySQL, developed in Perl. Percona Toolkit can be used to prove replication is working correctly, fix corrupted data, automate repetitive tasks, and speed up servers. Percona Toolkit is included with several Linux distributions such as CentOS and Debian, and packages are available for Fedora and Ubuntu as well. Percona Toolkit was originally developed as Maatkit, but as of late 2011, Maatkit is no longer developed.

jQuery

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It is free, open-source software using the permissive MIT license. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin.

jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

Figure 19 jQuery

The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its selector engine (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard Selectors API.

Microsoft and Nokia bundle jQuery on their platforms. Microsoft includes it with Visual Studio for use within Microsoft's ASP.NET AJAX and ASP.NET MVC frameworks while Nokia has integrated it into the Web Run-Time widget development platform.

jQuery, at its core, is a Document Object Model (DOM) manipulation library. The DOM is a tree-structure representation of all the elements of a Web page. jQuery simplifies the syntax for finding, selecting, and manipulating these DOM elements. For example, jQuery can be used for finding an element in the document with a certain property (e.g. all elements with an h1 tag), changing one or more of its attributes (e.g. color, visibility), or making it respond to an event (e.g. a mouse click).

jQuery also provides a paradigm for event handling that goes beyond basic DOM element selection and manipulation. The event assignment and the event callback function definition are done in a single step in a single location in the code. jQuery also aims to incorporate other highly used JavaScript functionality (e.g. fade ins and fade outs when hiding elements, animations by manipulating CSS properties).

The principles of developing with jQuery are:

- **Separation of JavaScript and HTML**: The jQuery library provides simple syntax for adding event handlers to the DOM using JavaScript, rather than

adding HTML event attributes to call JavaScript functions. Thus, it encourages developers to completely separate JavaScript code from HTML markup.
- **Brevity and clarity**: jQuery promotes brevity and clarity with features like chainable functions and shorthand function names.
- **Elimination of cross-browser incompatibilities**: The JavaScript engines of different browsers differ slightly so JavaScript code that works for one browser may not work for another. Like other JavaScript toolkits, jQuery handles all these cross-browser inconsistencies and provides a consistent interface that works across different browsers.
- **Extensibility**: New events, elements, and methods can be easily added and then reused as a plugin.

Features
- DOM element selections using the multi-browser open source selector engine Sizzle, a spin-off of the jQuery project
- DOM manipulation based on CSS selectors that uses elements' names and attributes, such as id and class, as criteria to select nodes in the DOM
- Events
- Effects and animations
- Ajax
- Deferred and Promise objects to control asynchronous processing
- JSON parsing
- Extensibility through plug-ins
- Utilities, such as feature detection
- Compatibility methods that are natively available in modern browsers, but need fall backs for older ones, such as inArray() and each()
- Multi-browser (not to be confused with cross-browser) support

Hardware Interfaces
- **Compatible OS**: Windows XP SP3+, Windows Vista SP2+, Windows Server 2003 SP2+, Windows 7, Windows Server 2008, Windows Server 2008 R2.
    - **Note*: You must have administrator privileges on your computer to run AMPPS.
- **Space**: Capacity of minimum 1.5GB Hard Disk space.
- **Memory**: 1GB RAM

Above are the bare minimum requirements for the WAMP stack we will be using on the server side to run Apache, PHP and MySQL.

Specific Requirements

Functions
- System login
- Creating Event
- Viewing Event details
- Editing Event details

- Inviting guests
- Creating checklists
- Adding Tasks
- Adding Images to event gallery
- Commenting on Images
- Adding Locations and Venues
- Booking Venues for events
- Assigning and Managing Tasks
- Validation of data
- Report generation

Performance Requirements

- Website should be able to handle 100,000 clients on the same time
- All the data should be inserted in an Atomic manner
- Downtime of server should be below 10 seconds

Above are the most basic requirements that should be met in any website of such crucial nature.

Data Models

DFD

Context Level Data Flow Diagram

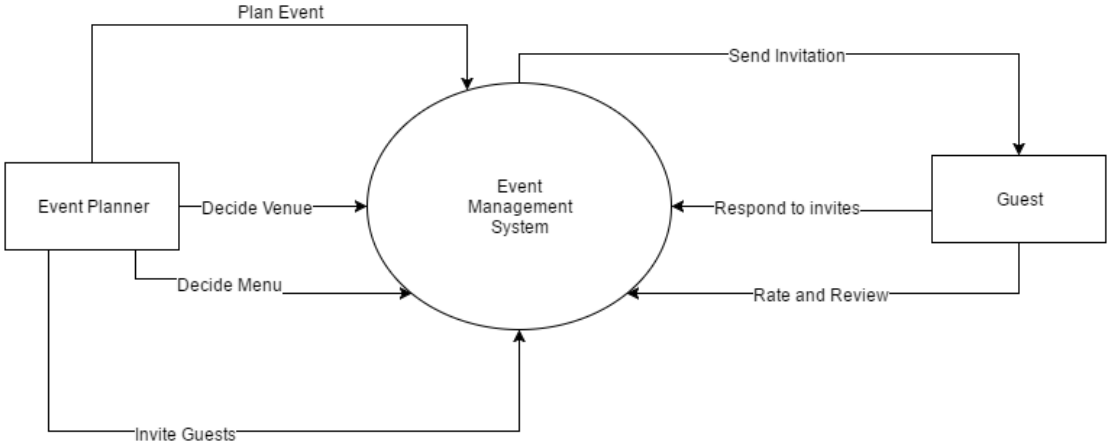


Figure 20 Context DFD

Data Flow while creating and managing an event

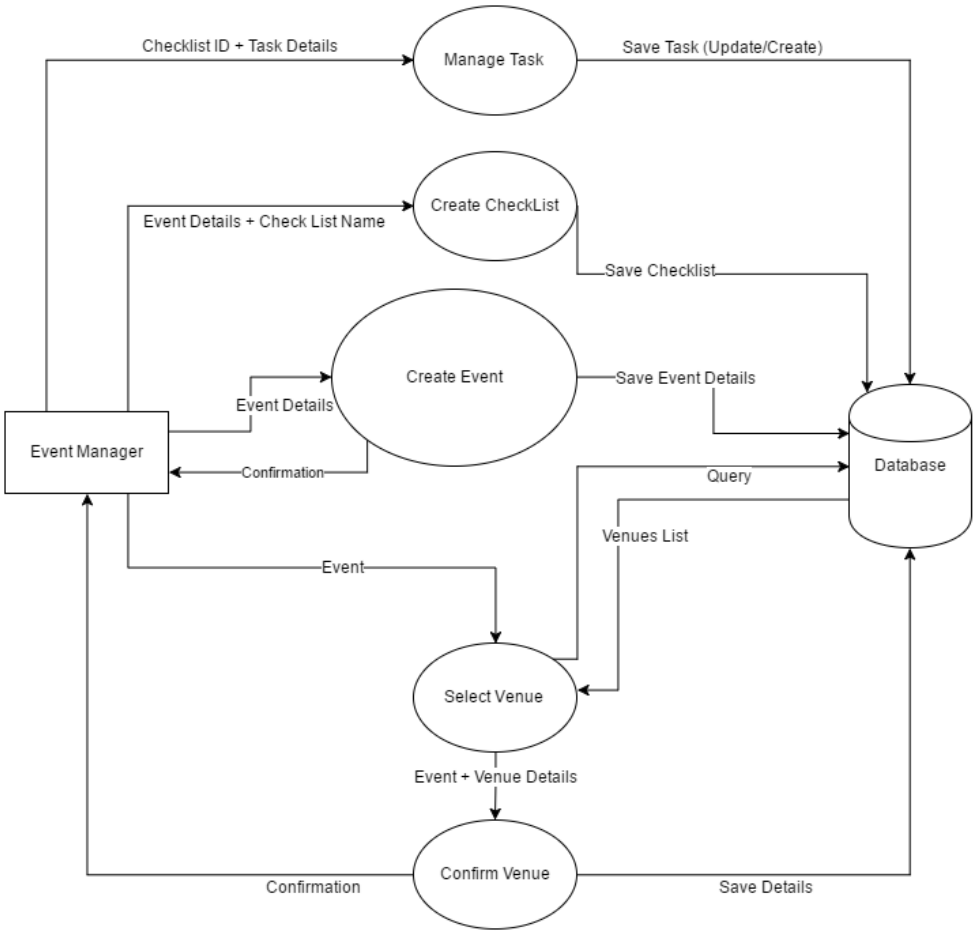


Figure 21 DFD Task Creation

Site Plans

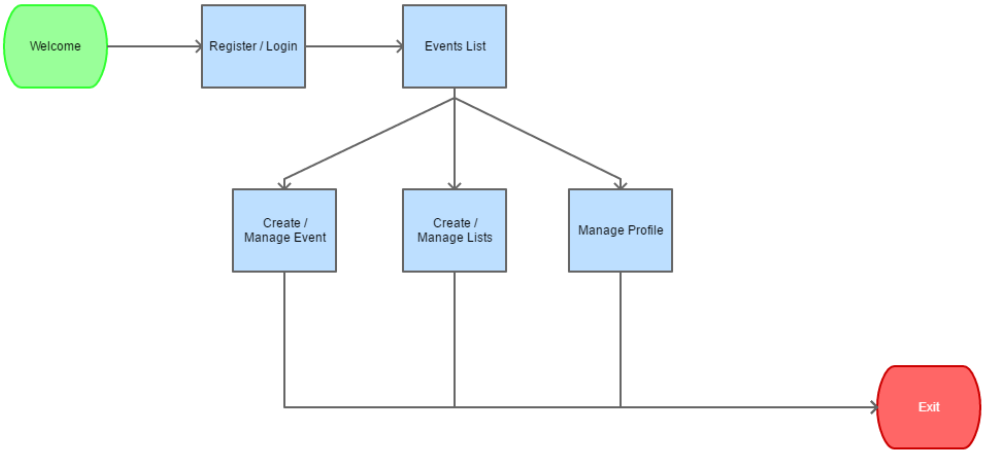


Figure 22 Base Plan

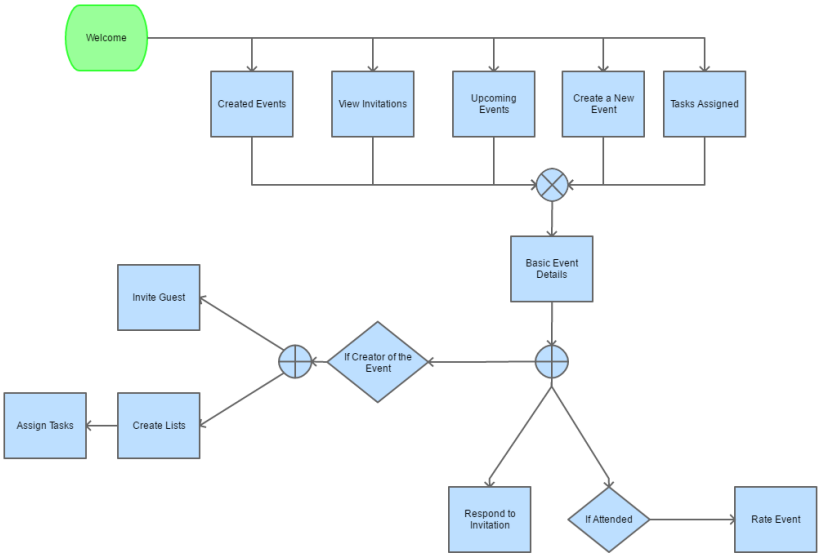


Figure 23 Event Management

Class Diagrams

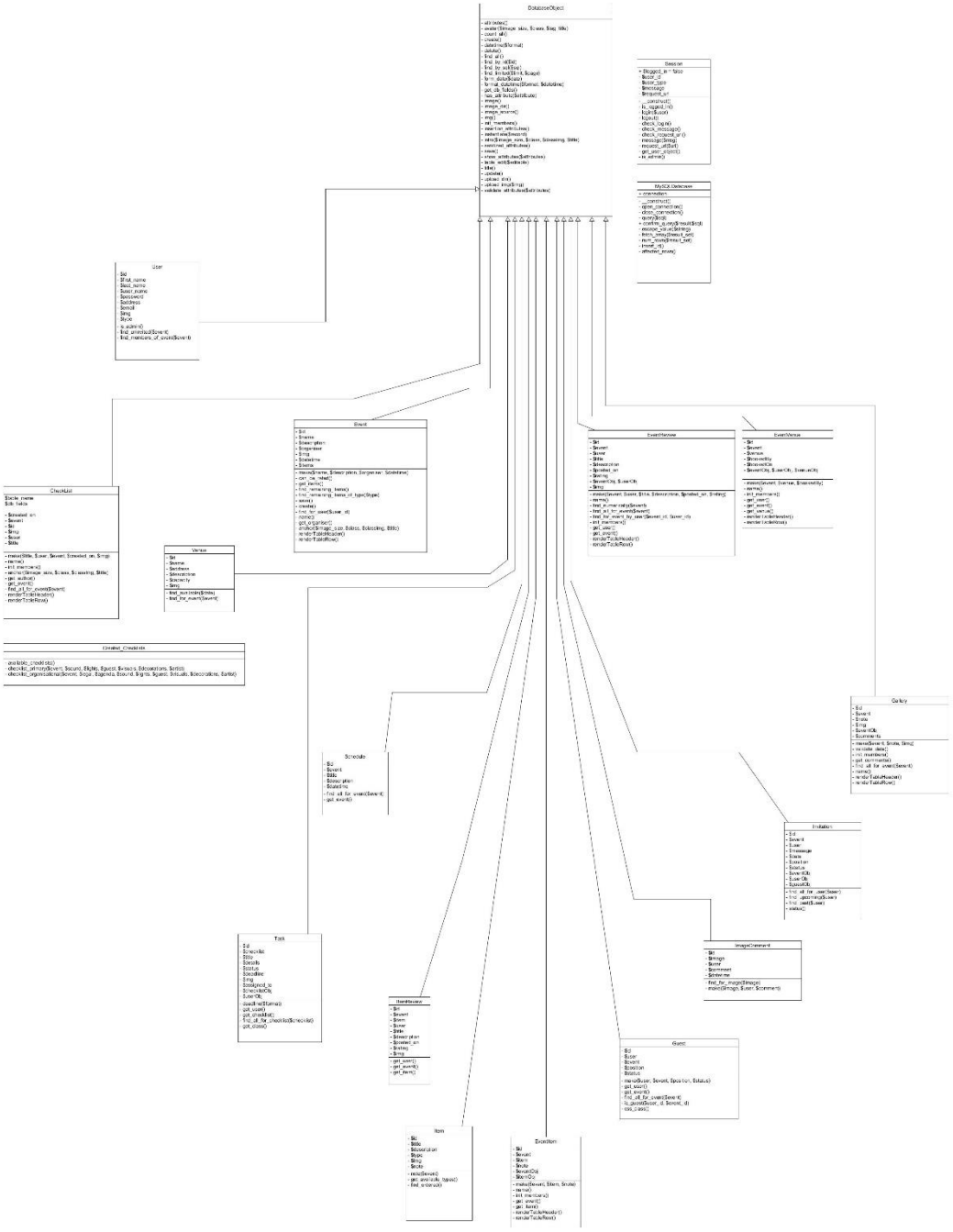


Figure 24 Class Diagram

Use Case Diagrams

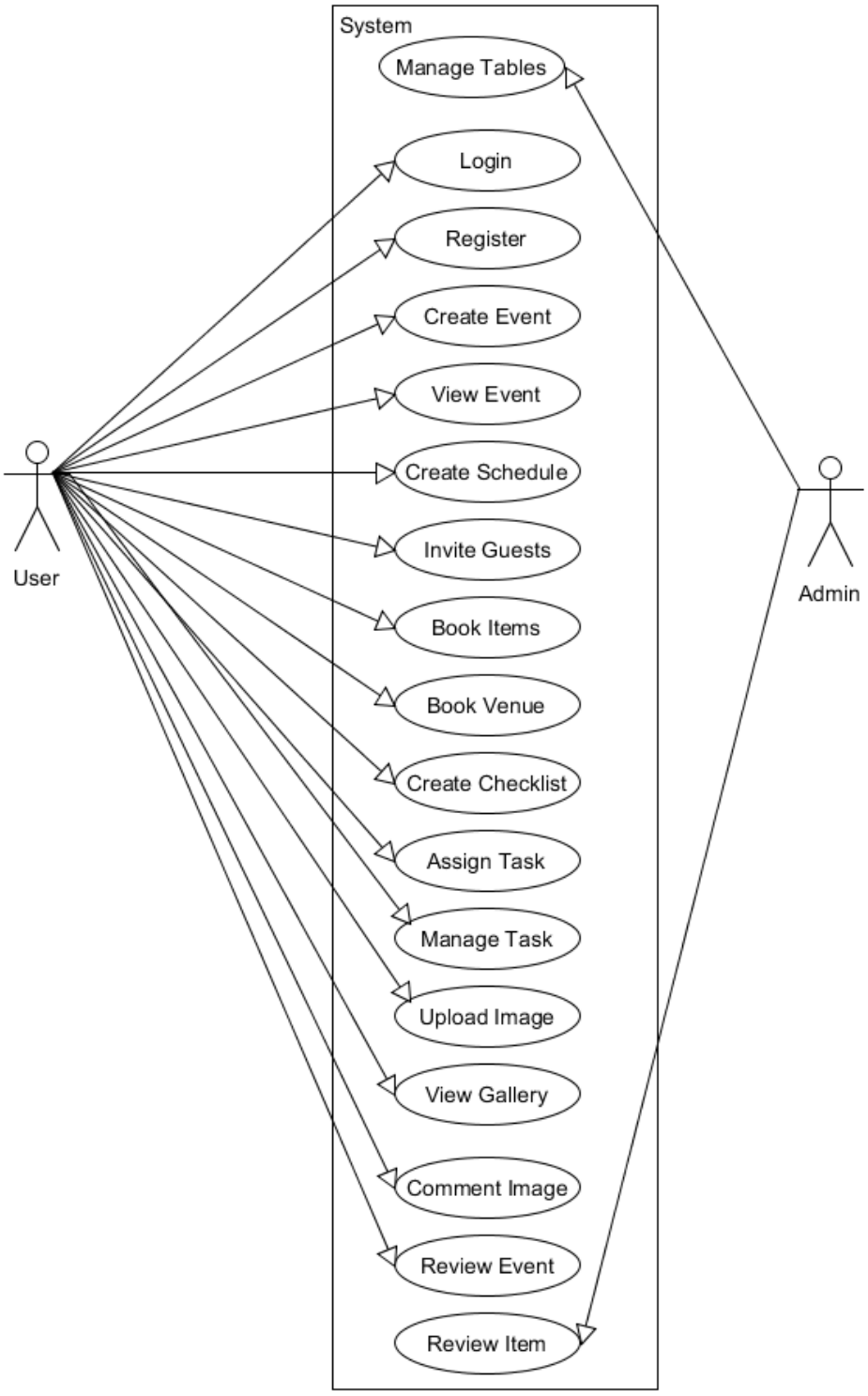


Figure 25 Use Case Diagram

System Design

Modularisation Details

Admin Module

There will be tasks in the project that require supervision. Such as to validate if the venue is accurately rated, or in fact available in the list. Same goes for user management tasks such as forgotten passwords, etc. So the admin module will cater to those needs and will have such tasks that only admins can perform. These admin account will be created by other admins or the super admin whose account is of the owner of the project.

But the major portion of the application will be from the perspective of the common user as an admin can also be required to perform such tasks.

Tables Management

This will allow admins to view all the database tables there are in the system and make sure that everything is in check. Also this same portal helps an admin in creating / editing / deleting.

- Users
- Events
- Tasks
- Checklists
- Venues
- Reviews
- So on…

User Modules

Following are the modules that **every user** of the project can operate onto. These are to be created in a highly cohesive and loosely coupled manner to ensure further applications and enhancements in later stages.

Event Creation

This will allow the users to create a new event with filling out a simple form that asks them to input the Event name, description, date and time.

Venue Decision

Every event is bound to its unique criteria and that includes the nature of event, number of guests and the mood event needs to have. All these aspects are valuable inputs for us to decide on the location and venue for an event. A suitable venue is something that can accommodate all the guests and gives the feel of event. Like we cannot organize a formal business meeting or seminar in a garden as the garden will make the event more of a casual gathering not a formal one.

Schedule

A good event manager always schedules all the small parts and performances etc for an event in a way that they do not interfere with each other and the guests are

entertained till the end. The app will help user to keep track of such things in an elegant timeline view that can be modified beforehand.

Guest List
Every event should have guest list to avoid problems like forgetting to invite someone to the event and later face the problems. This project helps a user keep track of all the guests and send invitations to them as well. On top of just sending the invite, the application also asks the guest to respond so the event manager can plan accordingly.

Catering and Menu
The application will help an event manager to plan for the menu of the whole event and keep track of it as well.

Custom Checklists
One can only predict a finite number of tasks in the vast concept of event management. There always are some unforeseen circumstances different for every event. To accommodate and handle them the application offers custom checklists.

Decorations list
Visuals play a vital role in our regular life, but for an event the decorations are more than that. Decorations and arrangements are the most important part of all the events. So the application also helps us to keep track of such arrangements.

Event Reviews
Any user who attended the particular event can later review the event so that the event planner can get insights on what can be done for future events of similar nature. Also by allowing users to review an event we are making the system more efficient.

Database Design

Database design is the process of producing a detailed data model of database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

Determine the data to be stored in the database.

Determine the relationships between the different data elements.

Superimpose a logical structure upon the data on the basis of these relationships.

Within the relational model the final step above can generally be broken down into two further steps that of determining the grouping of information within the system, generally determining what are the basic objects about which information is being stored, and then determining the relationships between these groups of information, or objects. This step is not necessary with an Object database.

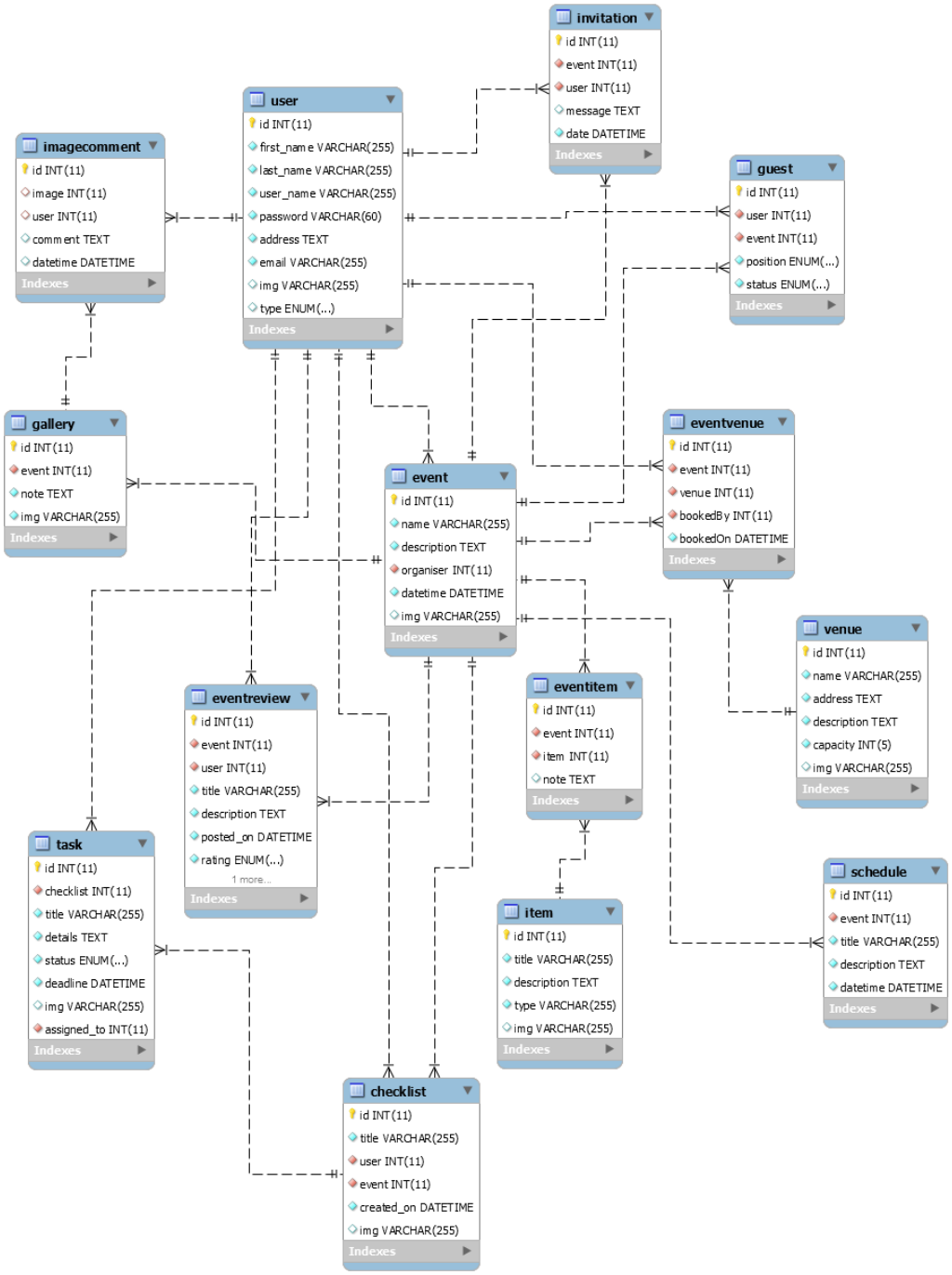


Figure 26 ER Diagram

User Interface Design

User interface design (UI) or user interface engineering is the design of user interfaces for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing usability and the user experience. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals (user centric design).

Good user interface design facilitates finishing the task at hand without drawing unnecessary attention to itself. Graphic design and typography are utilized to support its usability, influencing how the user performs certain interactions and improving the aesthetic appeal of the design; design aesthetics may enhance or detract from the ability of users to use the functions of the interface. The design process must balance technical functionality and visual elements (e.g., mental model) to create a system that is not only operational but also usable and adaptable to changing user needs.

Interface design is involved in a wide range of projects from computer systems, to cars, to commercial planes; all of these projects involve much of the same basic human interactions yet also require some unique skills and knowledge. As a result, designers tend to specialize in certain types of projects and have skills focused on their expertise, whether that be software design, user research, web design, or industrial design.

Test Cases

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

Typical test case parameters:

- Test Case ID
- Test Scenario
- Test Case Description
- Test Steps
- Prerequisite
- Test Data
- Expected Result
- Test Parameters
- Actual Result
- Environment Information
- Comments

Example

Scenario	Test Step	Expected Result	Actual Outcome
Verify that the input field that can accept maximum of 10 characters	Login to application and key in 10 characters	Application should be able to accept all 10 characters.	Application accepts all 10 characters.
Verify that the input field that can accept maximum of 11 characters	Login to application and key in 11 characters	Application should NOT accept all 11 characters.	Application accepts all 10 characters.

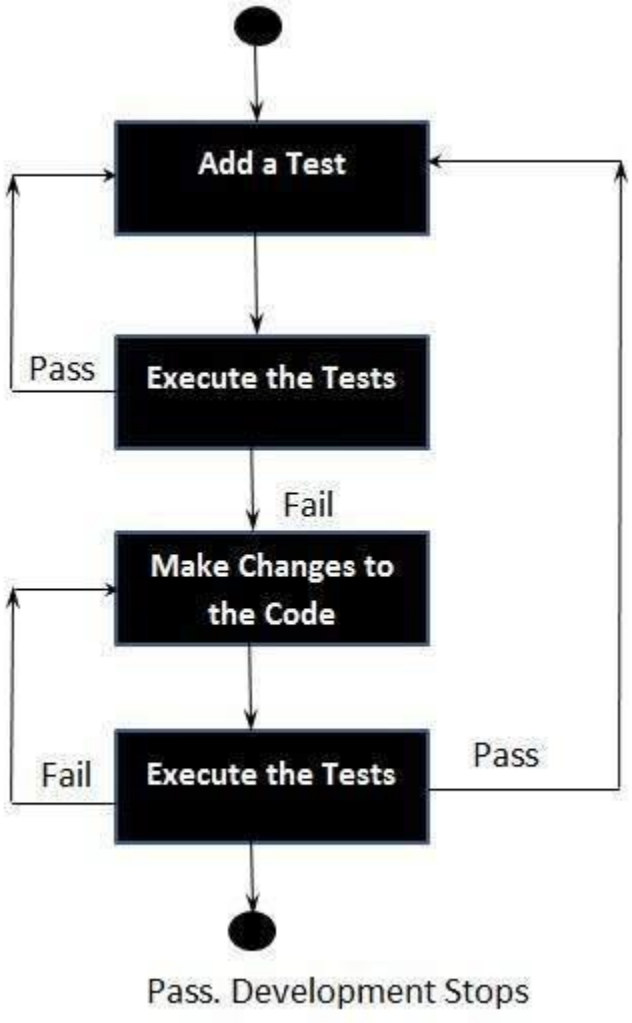


Figure 27 Test driven development process

Benefits of Test Driven Design Approach

- Much less debug time
- Code proven to meet requirements
- Tests become Safety Net
- Near zero defects
- Shorter development cycles

Base Test cases for this project

Test Scenario	Test Description	Input	Expected Outcome
User Login	Upon Input of Correct Username Password combination the user should be allowed to enter the system	Username Password	User Page for Logged in account
User Login	Upon Input of Correct Username Password combination the user should be allowed to enter the system	Username Password Incorrect	Incorrect Credentials Messsage
User Registeration	First time users fill out a simple form to gain access into the system	Personal Details Username Password	User Page for newly created Account
Event Creation	Fill out Event details and start organising event	Event Title, Date time, and other details	Event Page for newly created event
Guest Invitation	Select Users from within the list and send invitations to them	User List	Invitation in their account and their names in guest list
Checklist Creation	Create checklist for an event	Title and Event the checklist is for	Checklist item displayed in the event
Task Creation	Add a task into an checklist and assign the task to other members	Task title, description, assigned to, deadline	Task assignment to correct member and displayed in their portal

Image Upload	Upload an Image to event's gallery	Image	Image displayed in event's gallery
Image Comment	Comment on an image	Comment text	Comment shown in the comments panel of Image viewer under User's name who posted the comment
Logout	Exit the portal by ending session	Null	Secure pages require login again
Add Schedule	Select an event and add a schedule to it	New schedule's time	Schedule added in the event.

Coding

SQL commands

Database Creation

Table structure for table `checklist`

```
CREATE TABLE `checklist` (
  `id` int(11) NOT NULL,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `user` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `created_on` datetime NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `event`

```
CREATE TABLE `event` (
  `id` int(11) NOT NULL,
  `name` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `organiser` int(11) NOT NULL,
  `datetime` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `eventitem`

```
CREATE TABLE `eventitem` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `item` int(11) NOT NULL,
  `note` text
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Table structure for table `eventreview`

```
CREATE TABLE `eventreview` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `user` int(11) NOT NULL,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `posted_on` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `rating` enum('Poor','Below
Average','Average','Good','Great','Fantastic') COLLATE
utf8_unicode_ci NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `eventvenue`

```
CREATE TABLE `eventvenue` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `venue` int(11) NOT NULL,
  `bookedBy` int(11) NOT NULL,
  `bookedOn` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `gallery`

```
CREATE TABLE `gallery` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `note` text NOT NULL,
  `img` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Table structure for table `guest`

```
CREATE TABLE `guest` (
  `id` int(11) NOT NULL,
  `user` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `position` enum('Guest of
Honor','V.I.P','Guest','Member','Admin') COLLATE
utf8_unicode_ci NOT NULL DEFAULT 'Guest',
  `status` enum('Attending','Not Attending','May Be')
COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `imagecomment`

```
CREATE TABLE `imagecomment` (
  `id` int(11) UNSIGNED NOT NULL,
  `image` int(11) DEFAULT NULL,
  `user` int(11) DEFAULT NULL,
  `comment` text,
  `datetime` datetime DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Table structure for table `invitation`

```
CREATE TABLE `invitation` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `user` int(11) NOT NULL,
  `message` text,
  `date` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Table structure for table `item`

```
CREATE TABLE `item` (
  `id` int(11) NOT NULL,
```

```
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `type` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `itemreview`

```
CREATE TABLE `itemreview` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `item` int(11) NOT NULL,
  `user` int(11) NOT NULL,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `posted_on` datetime NOT NULL,
  `rating` enum('Poor','Below
Average','Average','Good','Great','Fantastic') COLLATE
utf8_unicode_ci NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `schedule`

```
CREATE TABLE `schedule` (
  `id` int(11) NOT NULL,
  `event` int(11) NOT NULL,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `datetime` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `task`

```
CREATE TABLE `task` (
  `id` int(11) NOT NULL,
  `checklist` int(11) NOT NULL,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `details` text COLLATE utf8_unicode_ci NOT NULL,
  `status`
enum('Assigned','Started','Working','Completed','Failed')
COLLATE utf8_unicode_ci NOT NULL DEFAULT 'Assigned',
  `deadline` datetime NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `assigned_to` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `user`

```
CREATE TABLE `user` (
  `id` int(11) NOT NULL,
```

```
  `first_name` varchar(255) COLLATE utf8_unicode_ci NOT
NULL,
  `last_name` varchar(255) COLLATE utf8_unicode_ci NOT
NULL,
  `user_name` varchar(255) COLLATE utf8_unicode_ci NOT
NULL,
  `password` varchar(60) COLLATE utf8_unicode_ci NOT NULL,
  `address` text COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `type` enum('admin','user','client') COLLATE
utf8_unicode_ci DEFAULT 'client'
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Table structure for table `venue`

```
CREATE TABLE `venue` (
  `id` int(11) NOT NULL,
  `name` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `address` text COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `capacity` int(5) NOT NULL,
  `img` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
```

Database indexing

Indexes for table `checklist`

```
ALTER TABLE `checklist`
  ADD PRIMARY KEY (`id`),
  ADD KEY `created_by_idx` (`user`),
  ADD KEY `created_for_idx` (`event`);
```

Indexes for table `event`

```
ALTER TABLE `event`
  ADD PRIMARY KEY (`id`),
  ADD KEY `organiser_idx` (`organiser`);
```

Indexes for table `eventitem`

```
ALTER TABLE `eventitem`
  ADD PRIMARY KEY (`id`),
  ADD KEY `selected_for_idx` (`event`),
  ADD KEY `selected_item_idx` (`item`);
```

Indexes for table `eventreview`

```
ALTER TABLE `eventreview`
  ADD PRIMARY KEY (`id`),
  ADD KEY `posted_by_idx` (`user`),
  ADD KEY `posted_for_idx` (`event`);
```

Indexes for table `eventvenue`

```
ALTER TABLE `eventvenue`
  ADD PRIMARY KEY (`id`),
  ADD KEY `booked_idx` (`venue`),
  ADD KEY `booked_for_idx` (`event`),
  ADD KEY `booked_by_idx` (`bookedBy`);
```

Indexes for table `gallery`

```
ALTER TABLE `gallery`
  ADD PRIMARY KEY (`id`),
  ADD KEY `clicked_for_idx` (`event`);

Indexes for table `guest`
ALTER TABLE `guest`
  ADD PRIMARY KEY (`id`),
  ADD KEY `invited_for_idx` (`event`),
  ADD KEY `guest_is_idx` (`user`);
```

Indexes for table `imagecomment`

```
ALTER TABLE `imagecomment`
  ADD PRIMARY KEY (`id`),
  ADD KEY `comment_for_idx` (`image`),
  ADD KEY `comment_by_idx` (`user`);
```

Indexes for table `invitation`

```
ALTER TABLE `invitation`
  ADD PRIMARY KEY (`id`),
  ADD KEY `invitation_for_idx` (`user`),
  ADD KEY `invited_event_idx` (`event`);
```

Indexes for table `item`

```
ALTER TABLE `item`
  ADD PRIMARY KEY (`id`);
```

Indexes for table `itemreview`

```
ALTER TABLE `itemreview`
  ADD PRIMARY KEY (`id`);
```

Indexes for table `schedule`

```
ALTER TABLE `schedule`
  ADD PRIMARY KEY (`id`),
  ADD KEY `schedule_for_idx` (`event`);
```

Indexes for table `task`

```
ALTER TABLE `task`
  ADD PRIMARY KEY (`id`),
  ADD KEY `belongs_to_idx` (`checklist`),
  ADD KEY `given_to_idx` (`assigned_to`);
```

Indexes for table `user`

```
ALTER TABLE `user`
```

```
  ADD PRIMARY KEY (`id`);
```

Indexes for table `venue`

```
ALTER TABLE `venue`
  ADD PRIMARY KEY (`id`);
```

Constraints

Constraints for table `checklist`

```
ALTER TABLE `checklist`
  ADD CONSTRAINT `created_by` FOREIGN KEY (`user`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `created_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `event`

```
ALTER TABLE `event`
  ADD CONSTRAINT `organiser` FOREIGN KEY (`organiser`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `eventitem`

```
ALTER TABLE `eventitem`
  ADD CONSTRAINT `selected_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `selected_item` FOREIGN KEY (`item`)
REFERENCES `item` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `eventreview`

```
ALTER TABLE `eventreview`
  ADD CONSTRAINT `posted_by` FOREIGN KEY (`user`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `posted_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `eventvenue`

```
ALTER TABLE `eventvenue`
  ADD CONSTRAINT `booked` FOREIGN KEY (`venue`) REFERENCES
`venue` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  ADD CONSTRAINT `booked_by` FOREIGN KEY (`bookedBy`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `booked_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `gallery`

```
ALTER TABLE `gallery`
  ADD CONSTRAINT `clicked_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `guest`

```
ALTER TABLE `guest`
  ADD CONSTRAINT `guest_is` FOREIGN KEY (`user`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `invited_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `imagecomment`

```
ALTER TABLE `imagecomment`
  ADD CONSTRAINT `comment_by` FOREIGN KEY (`user`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `comment_for` FOREIGN KEY (`image`)
REFERENCES `gallery` (`id`) ON DELETE NO ACTION ON UPDATE
NO ACTION;
```

Constraints for table `invitation`

```
ALTER TABLE `invitation`
  ADD CONSTRAINT `invitation_for` FOREIGN KEY (`user`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  ADD CONSTRAINT `invited_event` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `schedule`

```
ALTER TABLE `schedule`
  ADD CONSTRAINT `schedule_for` FOREIGN KEY (`event`)
REFERENCES `event` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Constraints for table `task`

```
ALTER TABLE `task`
  ADD CONSTRAINT `belongs_to` FOREIGN KEY (`checklist`)
REFERENCES `checklist` (`id`) ON DELETE NO ACTION ON
UPDATE NO ACTION,
  ADD CONSTRAINT `given_to` FOREIGN KEY (`assigned_to`)
REFERENCES `user` (`id`) ON DELETE NO ACTION ON UPDATE NO
ACTION;
```

Data Insertion

Insertion into Checklist

```
INSERT INTO `checklist` VALUES(1, 'Checklist 1', 3, 1,
'2017-03-25 01:35:41', '1490448967-6998-1489392994-8794-
LegendarY-wallpaper-10232789.jpg');
INSERT INTO `checklist` VALUES(2, 'Another List', 2, 2,
'2017-03-25 01:36:13', '1490448979-9528-1489564707-1364-
1543 - Clients.png');
INSERT INTO `checklist` VALUES(3, 'Food', 1, 1, '2017-03-
25 01:36:23', '1490449002-6550-f5.jpg');
INSERT INTO `checklist` VALUES(4, 'Beverages', 2, 1,
'2017-03-25 02:15:15', '1490451345-8498-dro.jpg');
INSERT INTO `checklist` VALUES(5, 'Study', 1, 2, '2017-03-
25 01:36:59', '1490449029-6344-p2.jpg');
INSERT INTO `checklist` VALUES(6, 'Custom List #6', 1, 1,
'2017-03-25 02:15:51', '1490451380-5674-chee.png');
INSERT INTO `checklist` VALUES(7, 'Custom List #7', 3, 4,
'2017-03-25 02:14:59', '1490451307-2969-1490426947-3986-
mickeyface.gif');
INSERT INTO `checklist` VALUES(8, 'Result', 2, 2, '2017-
03-25 01:38:02', '1490449091-9070-1489568768-6211-95 -
Cheque.png');
INSERT INTO `checklist` VALUES(9, 'Funding', 3, 5, '2017-
03-25 01:38:18', '1490449121-9389-1489653937-4158-Planet
Jupiter.png');
INSERT INTO `checklist` VALUES(10, 'Party Decorations', 1,
7, '2017-03-31 08:23:53', 'defaults\\party.png');
INSERT INTO `checklist` VALUES(11, 'Sound', 1, 8, '2017-
03-31 08:34:54', 'defaults\\microphone.png');
INSERT INTO `checklist` VALUES(12, 'Guest Management', 1,
8, '2017-03-31 08:34:54', 'defaults\\guests.png');
INSERT INTO `checklist` VALUES(13, 'Party Visuals', 1, 8,
'2017-03-31 08:34:54', 'defaults\\confetti.png');
INSERT INTO `checklist` VALUES(14, 'Party Decorations', 1,
8, '2017-03-31 08:34:54', 'defaults\\party.png');
INSERT INTO `checklist` VALUES(15, 'Artist', 1, 8, '2017-
03-31 08:34:54', 'defaults\\singer.png');
INSERT INTO `checklist` VALUES(16, 'Sound', 1, 9, '2017-
03-31 08:42:26', 'defaults\\microphone.png');
INSERT INTO `checklist` VALUES(17, 'Lights and
Electricity', 1, 9, '2017-03-31 08:42:26',
'defaults\\light-bulb.png');
INSERT INTO `checklist` VALUES(18, 'Sound', 1, 10, '2017-
03-31 08:59:07', 'defaults\\microphone.png');
INSERT INTO `checklist` VALUES(19, 'Lights and
Electricity', 1, 10, '2017-03-31 08:59:07',
'defaults\\light-bulb.png');
INSERT INTO `checklist` VALUES(20, 'Party Decorations', 1,
10, '2017-03-31 08:59:07', 'defaults\\party.png');
INSERT INTO `checklist` VALUES(21, 'Legal Docs', 1, 10,
'2017-03-31 08:59:07', 'defaults\\contract.png');
INSERT INTO `checklist` VALUES(22, 'Agenda', 1, 10, '2017-
03-31 08:59:07', 'defaults\\notebook.png');
```

Insertion into Event

```
INSERT INTO `event` VALUES(1, 'MCA Project', 'Qui tation
vivendum ex. Vim ex brute animal eloquentiam, assum summo
cum cu. Veritus delectus ea mel, et cum doctus laoreet, eu
congue latine omittam per. Recteque instructior usu ei,
has harum inimicus conceptam et, ne porro clita tollit
has.', 1, '2017-03-23 19:25:03', '1490449184-3108-party-
poster-retro-design_23-2147493281.jpg');
INSERT INTO `event` VALUES(2, 'Term end examinations', 'An
event that happens half yearly.', 2, '2017-03-29
13:31:00', '1490449218-4495-battle.jpg');
INSERT INTO `event` VALUES(3, 'Upload Image', 'An event of
uploading images on server.', 1, '2017-03-23 11:28:00',
'1490449251-6534-summer.jpg');
INSERT INTO `event` VALUES(4, 'Testing images', 'Hello
there', 2, '2017-03-23 15:25:03', '1490449297-3983-
foodfe.jpg');
INSERT INTO `event` VALUES(5, 'Defiance', 'Legacy of kain
is one thing but Raziel will prevail always.', 3, '2017-
03-10 11:46:00', '1490449336-9306-ju.jpg');
INSERT INTO `event` VALUES(6, 'Soul Reaver 2', 'Travelling
to NOSGOTH to meet Raziel and Mobius.\r\n\r\nLorem ipsum
dolor sit amet, consectetur adipiscing elit. Maecenas
elementum rutrum velit eget pretium. Integer vel dui quis
arcu euismod facilisis sed non purus. Donec at nisi purus.
Pellentesque ut laoreet sapien. Sed eget est interdum,
tempor metus commodo, interdum diam. Nam et mauris metus.
Nunc congue eleifend orci, id.', 3, '2017-03-10 11:46:00',
'1490449357-5741-brree.jpg');
INSERT INTO `event` VALUES(7, 'Party on 4th', 'Party all
night', 1, '2017-04-04 20:00:00', '1490948633-9582-
party_all.jpg');
INSERT INTO `event` VALUES(8, 'Party on 11th', 'Party all
night', 1, '2017-04-11 14:04:00', '1490949294-2547-
party_all.jpg');
INSERT INTO `event` VALUES(9, 'April Fools Party', 'The
party where nobody is a fool but still all are. As there
will be no party there....', 1, '2017-04-01 20:15:00',
'1490949746-2345-april_fool.jpg');
INSERT INTO `event` VALUES(10, 'Closing Day', 'Keep record
for financial year and make decisions for next year. Also
have a little bit of party for everyone\'s motivation.',
1, '2017-03-31 23:59:00', '1490950747-8488-book.jpg');
```

Insert into EventItem

```
INSERT INTO `eventitem` VALUES(1, 1, 1, 'Yummy starters
for everyone. <strong>On the house</strong>');
INSERT INTO `eventitem` VALUES(2, 1, 2, 'This is the best!
<em>Everyone is gonna love it</em>');
INSERT INTO `eventitem` VALUES(3, 8, 1, 'Pre selected in
package.');
```

```
INSERT INTO `eventitem` VALUES(4, 8, 4, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(5, 8, 2, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(6, 9, 5, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(7, 9, 7, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(8, 9, 2, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(9, 10, 1, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(10, 10, 4, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(11, 10, 2, 'Pre selected in
package.');
INSERT INTO `eventitem` VALUES(12, 10, 3, 'Pre selected in
package.');
```

Insert into EventReview

```
INSERT INTO `eventreview` VALUES(1, 5, 3, 'It really was
awesome', 'Lorem ipsum dolor sit amet, netus et malesuada
fames ac turpis egestas. Vestibulum tortor quam, feugiat
vitae, ultricies eget, tempor sit amet, ante. Donec eu
libero sit amet quam egestas semper.', '2017-03-25
01:42:53', 'Good', '1490449409-9265-car-1.jpg');
INSERT INTO `eventreview` VALUES(2, 5, 2, 'Well Done', 'I
am preparing for exams as well.', '2017-03-25 01:43:35',
'Fantastic', '1490449448-7927-car-2.gif-c200');
INSERT INTO `eventreview` VALUES(3, 3, 1, 'A helping
hand', 'PHP is a lot of fun.', '2017-03-25 01:44:14',
'Average', '1490449484-4971-gold.jpg');
INSERT INTO `eventreview` VALUES(4, 5, 3, 'Yaes',
'Hellow', '2017-03-25 01:44:47', 'Fantastic', '1490449522-
5255-pip.jpg');
INSERT INTO `eventreview` VALUES(5, 5, 4, 'LaLALA',
'Hehe', '2017-03-25 01:45:25', 'Good', '1490449559-1352-
hurray.gif');
INSERT INTO `eventreview` VALUES(6, 5, 5, 'Rated', 'NO',
'2017-03-25 01:46:02', 'Below Average', '1490449604-7884-
mae.jpg');
INSERT INTO `eventreview` VALUES(7, 5, 1, 'Raziel is the
best', 'Being modest! Am I?', '2017-03-25 02:28:48',
'Average', '1490449632-5901-toad.png');
```

Insert into EventVenue

```
INSERT INTO `eventvenue` VALUES(1, 2, 1, 1, '2017-03-03
12:24:41');
INSERT INTO `eventvenue` VALUES(2, 1, 1, 2, '2017-03-03
02:08:00');
INSERT INTO `eventvenue` VALUES(3, 6, 2, 3, '2017-03-13
08:49:02');
```

```
INSERT INTO `eventvenue` VALUES(5, 6, 1, 1, '2017-03-13
08:51:57');
INSERT INTO `eventvenue` VALUES(6, 5, 1, 1, '2017-03-03
12:55:06');
```

Insert into Gallery

```
INSERT INTO `gallery` VALUES(1, 1, 'Hallway! Gate.',
'1490449831-4787-Phoenix_Auditorium.jpg');
INSERT INTO `gallery` VALUES(2, 1, 'It was awesome!!!',
'1490449879-7282-ev1.jpg');
INSERT INTO `gallery` VALUES(3, 1, 'Croud was great!',
'1490449905-9638-ev2.jpg');
INSERT INTO `gallery` VALUES(4, 1, 'A beautiful image of
event!', '1490449948-4604-ev3.jpg');
INSERT INTO `gallery` VALUES(5, 3, 'Butter Butter Butter
fly!', '1490450723-9922-ev4.jpg');
INSERT INTO `gallery` VALUES(6, 3, 'Autumn has come!',
'1490450778-5013-ev6.jpg');
INSERT INTO `gallery` VALUES(7, 3, 'Autumn has come! A
beautiful one!', '1490450803-4557-ev7.jpg');
```

Insert into Guest

```
INSERT INTO `guest` VALUES(1, 2, 1, 'Guest of Honor',
'Attending');
INSERT INTO `guest` VALUES(2, 3, 2, 'Guest of Honor',
'Attending');
INSERT INTO `guest` VALUES(3, 1, 1, 'Admin', 'Attending');
INSERT INTO `guest` VALUES(4, 3, 4, 'Member', 'Not
Attending');
INSERT INTO `guest` VALUES(5, 4, 3, 'Member', 'May Be');
INSERT INTO `guest` VALUES(6, 5, 2, 'Guest', 'May Be');
INSERT INTO `guest` VALUES(7, 1, 3, 'V.I.P', 'Not
Attending');
INSERT INTO `guest` VALUES(8, 1, 5, 'Guest of Honor',
'Attending');
INSERT INTO `guest` VALUES(9, 4, 1, 'Member', 'Not
Attending');
INSERT INTO `guest` VALUES(10, 6, 1, 'Guest', 'May Be');
INSERT INTO `guest` VALUES(11, 5, 1, 'V.I.P', 'May Be');
INSERT INTO `guest` VALUES(12, 3, 3, 'Member', 'Not
Attending');
INSERT INTO `guest` VALUES(13, 5, 3, 'Member', 'May Be');
INSERT INTO `guest` VALUES(14, 7, 3, 'Member', 'May Be');
INSERT INTO `guest` VALUES(15, 3, 5, 'Admin',
'Attending');
INSERT INTO `guest` VALUES(16, 2, 3, 'Guest of Honor',
'May Be');
INSERT INTO `guest` VALUES(17, 6, 5, 'Guest', 'May Be');
INSERT INTO `guest` VALUES(18, 8, 5, 'Guest', 'May Be');
INSERT INTO `guest` VALUES(19, 2, 4, 'V.I.P',
'Attending');
INSERT INTO `guest` VALUES(20, 1, 4, 'Member',
'Attending');
```

```
INSERT INTO `guest` VALUES(21, 2, 2, 'Admin',
'Attending');
INSERT INTO `guest` VALUES(22, 3, 1, 'Guest', 'May Be');
INSERT INTO `guest` VALUES(23, 3, 6, 'Admin', 'May Be');
INSERT INTO `guest` VALUES(24, 1, 7, 'Admin',
'Attending');
INSERT INTO `guest` VALUES(25, 1, 8, 'Admin',
'Attending');
INSERT INTO `guest` VALUES(26, 1, 9, 'Admin',
'Attending');
INSERT INTO `guest` VALUES(27, 1, 10, 'Admin',
'Attending');
```

Insert into ImageComment

```
INSERT INTO `imagecomment` VALUES(1, 1, 1, 'My Hallway is
tremendously big.', '2017-03-25 22:27:27');
INSERT INTO `imagecomment` VALUES(2, 1, 1, 'Ho HO HO!!',
'2017-03-25 23:00:27');
INSERT INTO `imagecomment` VALUES(3, 1, 2, 'The venue was
quite nice... I just loved it. Looking forward to do more
events like this.', '2017-03-25 23:01:39');
INSERT INTO `imagecomment` VALUES(4, 1, 2, 'Can you tell
when next event will be?', '2017-03-25 23:04:24');
INSERT INTO `imagecomment` VALUES(5, 1, 3, 'Yes please!',
'2017-03-25 23:05:49');
INSERT INTO `imagecomment` VALUES(6, 1, 3, 'I am
waiting...', '2017-03-25 23:06:50');
```

Insert into Invitation

```
INSERT INTO `invitation` VALUES(1, 1, 2, 'You are
requested to visit.', '2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(2, 2, 3, 'Hello there!!!',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(3, 1, 1, 'Please Come',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(4, 4, 3, 'Coming right?',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(5, 3, 4, 'Welcome!',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(6, 2, 5, 'Ho HO HO
OHOHOHOH', '2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(7, 3, 1, 'Will you?',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(8, 5, 1, 'Please visit and
play with us.', '2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(9, 1, 4, 'You are allowed
to visit and complete the projects. Please be on time and
you can always bring a +1 with you.', '2017-03-11
12:13:21');
INSERT INTO `invitation` VALUES(10, 1, 6, 'You are allowed
to visit and complete the projects. Please be on time and
you can always bring a +1 with you.', '2017-03-11
12:13:21');
```

```
INSERT INTO `invitation` VALUES(11, 1, 5, 'You are the
legend!! TEKKEN 7', '2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(12, 3, 3, 'This is a
testing event to find if users visit and enjoy.', '2017-
03-11 12:13:21');
INSERT INTO `invitation` VALUES(13, 3, 5, 'This is a
testing event to find if users visit and enjoy.', '2017-
03-11 12:13:21');
INSERT INTO `invitation` VALUES(14, 3, 7, 'This is a
testing event to find if users visit and enjoy.', '2017-
03-11 12:13:21');
INSERT INTO `invitation` VALUES(15, 5, 3, 'You created the
event!', '2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(16, 3, 2, 'Welcome!!',
'2017-03-11 12:13:21');
INSERT INTO `invitation` VALUES(17, 5, 6, 'Please come to
the event. We all are waiting for you.', '2017-03-13
09:05:14');
INSERT INTO `invitation` VALUES(18, 5, 8, 'Please come to
the event. We all are waiting for you.', '2017-03-13
09:05:14');
INSERT INTO `invitation` VALUES(19, 4, 2, 'Will you come?
Please??', '2017-03-13 09:11:41');
INSERT INTO `invitation` VALUES(20, 4, 1, 'Chup chaap aa
jaa!!!!', '2017-03-13 09:12:33');
INSERT INTO `invitation` VALUES(21, 2, 2, 'You are the
organizer!', '2017-03-15 07:52:35');
INSERT INTO `invitation` VALUES(22, 1, 3, 'You are
valuable to us!', '2017-03-15 09:07:36');
INSERT INTO `invitation` VALUES(23, 6, 3, 'You organized
the event!', '2017-03-15 09:27:33');
INSERT INTO `invitation` VALUES(24, 7, 1, 'You created the
event!', '2017-03-31 08:23:53');
INSERT INTO `invitation` VALUES(25, 8, 1, 'You created the
event!', '2017-03-31 08:34:54');
INSERT INTO `invitation` VALUES(26, 9, 1, 'You created the
event!', '2017-03-31 08:42:26');
INSERT INTO `invitation` VALUES(27, 10, 1, 'You created
the event!', '2017-03-31 08:59:07');
```

Insert into Item

```
INSERT INTO `item` VALUES(1, 'Pasta', 'Yummy Pasta for
starters.', 'Chineese', '1490450857-5255-pasta.jpg');
INSERT INTO `item` VALUES(2, 'Mc Aloo Tikki', 'A great
meal.', 'Fast Food', '1490450879-3780-mcall.jpg');
INSERT INTO `item` VALUES(3, 'Mc Wrap', 'Lorem ipsum dolor
sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut
enim ad minim veniam.', 'Fast Food', '1490450910-7778-
mcwra.jpg');
INSERT INTO `item` VALUES(4, 'Ramen', 'Pasta of china.
;)', 'Chineese', '1490450943-8629-ramen.jpg');
```

```
INSERT INTO `item` VALUES(5, 'Donalad Duck', 'Quack Quack
Quack, the signature.', 'Disney', '1490451034-9249-
don.jpg');
INSERT INTO `item` VALUES(6, 'Chip\'n\'Dale', 'Rescue
Rangers', 'Disney', '1490451066-4123-chip.jpg');
INSERT INTO `item` VALUES(7, 'Ariel', 'The little mermaid.
Aquatic themed party elements.', 'Disney', '1490451115-
3259-ariel.jpg');
```

Insert into ItemReview

```
INSERT INTO `itemreview` VALUES(1, 1, 1, 1, 'Good', 'It
was really yummy!!', '2017-03-25 02:12:04', 'Fantastic',
'1490451191-9840-lv.jpg');
INSERT INTO `itemreview` VALUES(2, 2, 1, 2, 'Not so good',
'I really can\'t say', '2017-03-25 02:13:14', 'Average',
'1490451206-6669-1000.jpg');
```

Insert into Schedule

```
INSERT INTO `schedule` VALUES(1, 1, 'Synopsis',
'Submitting the synopsis for project before 31st December
is a must.', '2016-11-20 11:00:00');
INSERT INTO `schedule` VALUES(2, 1, 'Project Design',
'Creating design for a project helps to know what to do,
when to do and how to do.', '2016-11-25 08:15:00');
INSERT INTO `schedule` VALUES(3, 3, 'Checks', 'Test test
and TEST', '2017-03-03 14:26:00');
INSERT INTO `schedule` VALUES(4, 1, 'Another task',
'Create and edit and modify', '2017-03-12 12:24:00');
INSERT INTO `schedule` VALUES(5, 6, 'Induction', 'Welcome
all the guests and see if they are up for the journey..',
'2017-10-03 12:00:00');
INSERT INTO `schedule` VALUES(6, 6, 'Tally Guests', 'Hello
Hello mic testing.', '2017-03-10 14:36:00');
```

Insert into Task

```
INSERT INTO `task` VALUES(1, 1, 'Bring Study Material',
'It is a must have.', 'Assigned', '2017-03-25 01:23:45',
'1490451281-8060-ts1.jpg', 1);
INSERT INTO `task` VALUES(2, 4, 'New Created Task', 'Do it
DO IT CSS', 'Working', '2017-03-04 11:13:00', '1490451290-
4455-1489646774-8179-93 - Record Clearance.png', 2);
INSERT INTO `task` VALUES(3, 1, 'Another task
calculation', 'Do it complete and get rewarded.',
'Completed', '2017-03-10 15:15:00', '1490451436-4473-
bv1.jpg', 1);
INSERT INTO `task` VALUES(4, 9, 'Go to bank', 'Go and find
all the details for loan process.', 'Completed', '2017-03-
11 15:15:00', '1490451451-9479-1489568976-6422-1553 -
Lock.png', 3);
INSERT INTO `task` VALUES(5, 9, 'Find Manager', 'If he
helps!', 'Completed', '2017-03-11 15:40:00', '1490451480-
1828-per.jpg', 3);
```

```
INSERT INTO `task` VALUES(6, 7, 'A new task', 'Lorem ipsum
dolor sit amet, consectetur adipiscing elit. Aliquam
molestie odio ut massa vehicula viverra. Nulla a enim
massa. Proin semper, neque vitae laoreet volutpat, dolor
elit dapibus urna, id pulvinar lectus dui vitae ligula.',
'Assigned', '2017-03-16 12:15:00', '1490451507-3557-
limie.jpg', 1);
INSERT INTO `task` VALUES(7, 7, 'Another task', 'Nam
condimentum odio faucibus tellus vestibulum et convallis
enim consectetur. Etiam lacinia ornare orci aliquam
pretium.', 'Assigned', '2017-03-16 13:15:00', '1490451531-
5185-c.jpg', 2);
INSERT INTO `task` VALUES(8, 1, 'New Task', 'A new task
just to ensure the aside works.', 'Assigned', '2017-03-31
23:59:00', '1490451550-1641-te.jpg', 1);
INSERT INTO `task` VALUES(9, 11, 'Arrange Mics', 'You
should find if they can be used.', 'Assigned', '2017-04-10
00:00:00', '1491024345-6189-microphone.png', 1);
```

Insert into User

```
INSERT INTO `user` VALUES(1, 'Lakshay', 'Verma',
'lakshay', '8946553', 'Jalandhar Cantt',
'verma_lakshay@live.in', '1490450106-6262-lk.jpg',
'admin');
INSERT INTO `user` VALUES(2, 'Mandeep', 'Kaur', 'mandeep',
'1234567', 'Jalandhar', 'mandeepshabina@gmail.com',
'1490448145-4361-1489396258-4222-Anime_Girl-wallpaper-
9816582.jpg', 'admin');
INSERT INTO `user` VALUES(3, 'Raziel', 'Sarafan',
'Raziel', '123456', 'Nosgoth', 'raziel@outlook.in',
'1490451605-7488-lok.png', 'admin');
INSERT INTO `user` VALUES(4, 'Dastan', 'Timeer', 'Dastan',
'123456', 'Persia', 'lakshayverma@outlook.in',
'1490448482-1468-1488528010-2486-cyanogenmod logo.png',
'admin');
INSERT INTO `user` VALUES(5, 'Heihachi', 'Mishima',
'heihachi', 'tekken', 'Japan', 'heihachi@tekken.com',
'1490448615-9277-mario.jpg', 'admin');
INSERT INTO `user` VALUES(6, 'Jhon', 'Constantine',
'jhonny', 'lalala', 'New Orleans',
'jhonny.boy@constantine.us', '1490450280-3102-dead.png',
'admin');
INSERT INTO `user` VALUES(7, 'Luffy', 'Monkey', 'hancock',
'123456', 'Grand Line', 'luffy@strawhats.jp', '1490448660-
4437-main-qimg-c0f2e7c8e5fb40c52acd389e5de0d314.png',
'admin');
INSERT INTO `user` VALUES(8, 'Boa', 'Hancock', 'hanboa',
'987654', 'Amazon', 'boa@op.com', '1490448716-9016-
boa.jpg', 'admin');
```

Insert into Venue

```
INSERT INTO `venue` VALUES(1, 'Vegas Paladium', 'House No
8 Mohalla No 14\r\nJalandhar Cantt', 'Lorem ipsum dolor
```

```
sit amet, consectetur adipiscing elit. Maecenas elementum
rutrum velit eget pretium. Integer vel dui quis arcu
euismod facilisis sed non purus. Donec at nisi purus.
Pellentesque ut laoreet sapien. Sed eget est interdum,
tempor metus commodo, interdum diam. Nam et mauris
metus.', 85, '1490448816-9151-vegas.jpg');
INSERT INTO `venue` VALUES(2, 'Nevada Consertium', 'Mel
ullum vulputate eu. fugit timeam lucilius ius in. Aliquam
maluisset sea at, ius at feugiat alienum.', 'Mea ad inani
aeque interesset, or ad vel. Mei ad audiam denique, et usu
soluta cetero reprehendunt.', 150, '1490448869-5649-
dinner.jpg');
INSERT INTO `venue` VALUES(3, 'Next Gen Hall', 'Aliquam
rhoncus magna nec elit placerat commodo. Aenean non ipsum
sit amet dolor lacinia blandit. Nam condimentum odio
faucibus tellus vestibulum et convallis enim.', 'Cum
sociis elementum pretium. Vivamus consectetur luctus dui
sit amet ultrices.', 150, '1490448921-4705-room.jpg');
```

Tables as in Database

Checklist

```
SELECT * FROM `checklist`
```

id	title	user	event	created_on	img
1	Checklist 1	3	1	2017-03-25 01:35:41	1490448967-6998-1489392994-8794-LegendarY-wallpape...
2	Another List	2	2	2017-03-25 01:36:13	1490448979-9528-1489564707-1364-1543 - Clients.png
3	Food	1	1	2017-03-25 01:36:23	1490449002-6550-f5.jpg
4	Beverages	2	1	2017-03-25 02:15:15	1490451345-8498-dro.jpg
5	Study	1	2	2017-03-25 01:36:59	1490449029-6344-p2.jpg
6	Custom List #6	1	1	2017-03-25 02:15:51	1490451380-5674-chee.png
7	Custom List #7	3	4	2017-03-25 02:14:59	1490451307-2969-1490426947-3986-mickeyface.gif
8	Result	2	2	2017-03-25 01:38:02	1490449091-9070-1489568768-6211-95 - Cheque.png
9	Funding	3	5	2017-03-25 01:38:18	1490449121-9389-1489653937-4158-Planet Jupiter.png
10	Party Decorations	1	7	2017-03-31 08:23:53	defaults\party.png
11	Sound	1	8	2017-03-31 08:34:54	defaults\microphone.png
12	Guest Management	1	8	2017-03-31 08:34:54	defaults\guests.png
13	Party Visuals	1	8	2017-03-31 08:34:54	defaults\confetti.png
14	Party Decorations	1	8	2017-03-31 08:34:54	defaults\party.png
15	Artist	1	8	2017-03-31 08:34:54	defaults\singer.png
16	Sound	1	9	2017-03-31 08:42:26	defaults\microphone.png
17	Lights and Electricity	1	9	2017-03-31 08:42:26	defaults\light-bulb.png
18	Sound	1	10	2017-03-31 08:59:07	defaults\microphone.png
19	Lights and Electricity	1	10	2017-03-31 08:59:07	defaults\light-bulb.png
20	Party Decorations	1	10	2017-03-31 08:59:07	defaults\party.png
21	Legal Docs	1	10	2017-03-31 08:59:07	defaults\contract.png
22	Agenda	1	10	2017-03-31 08:59:07	defaults\notebook.png
23	Sound	1	11	2017-04-23 02:55:43	defaults\microphone.png
24	Lights and Electricity	1	11	2017-04-23 02:55:43	defaults\light-bulb.png
25	Party Decorations	1	11	2017-04-23 02:55:43	defaults\party.png

Figure 28 Checklists Data

Event

```
SELECT * FROM `event`
```

id	name	description	organiser	datetime	img
1	MCA Project	Qui tation vivendum ex. Vim ex brute animal eloque...	1	2017-03-23 19:25:03	1490449184-3108-party-poster-retro-design_23-21474...
2	Term end examinations	An event that happens half yearly.	2	2017-03-29 13:31:00	1490449218-4495-battle.jpg
3	Upload Image	An event of uploading images on server.	1	2017-03-23 11:28:00	1490449251-6534-summer.jpg
4	Testing images	Hello there	2	2017-03-23 15:25:03	1490449297-3983-foodfe.jpg
5	Defiance	Legacy of kain is one thing but Raziel will prevai...	3	2017-03-10 11:46:00	1490449336-9306-ju.jpg
6	Soul Reaver 2	Travelling to NOSGOTH to meet Raziel and Mobius. ...	3	2017-03-10 11:46:00	1490449357-5741-brree.jpg
7	Party on 4th	Party all night	1	2017-04-04 20:00:00	1490948633-9582-party_all.jpg
8	Party on 11th	Party all night	1	2017-04-11 14:04:00	1490949294-2547-party_all.jpg
9	April Fools Party	The party where nobody is a fool but still all are...	1	2017-04-01 20:15:00	1490949746-2345-april_fool.jpg
10	Closing Day	Keep record for financial year and make decisions ...	1	2017-03-31 23:59:00	1490950747-8488-book.jpg
11	Project Submission	The event will make sure everyone submits their pr...	1	2017-04-29 00:00:00	1492916142-6931-nl3.jpg

Figure 29 Event Data

Event Item

```
SELECT * FROM `eventitem`
```

id	event	item	note
1	1	1	Yummy starters for everyone. On the house<...
2	1	2	This is the best! Everyone is gonna love it</e...
3	8	1	Pre selected in package.
4	8	4	Necessary for the event
5	8	2	Pre selected in package.
6	9	5	Pre selected in package.
7	9	7	Good option for the event
8	9	2	Pre selected in package.
9	10	1	Pre selected in package.
10	10	4	A must have.
11	10	2	Trying is must.
12	10	3	Pre selected in package.
13	11	1	Admin recommended.
14	11	5	Pre selected in package.
15	11	7	Pre selected in package.

Figure 30 Event Item Data

Event Review

```
SELECT * FROM `eventreview`
```

id	event	user	title	description	posted_on	rating	img
1	5	3	It really was awesome	Lorem ipsum dolor sit amet, netus et malesuada fam...	2017-03-25 01:42:53	Good	1490449409-9265-car-1.jpg
2	5	2	Well Done	I am preparing for exams as well.	2017-03-25 01:43:35	Fantastic	1490449448-7927-car-2.gif-c200
3	3	1	A helping hand	PHP is a lot of fun.	2017-03-25 01:44:14	Average	1490449484-4971-gold.jpg
4	5	3	Yaes	Hellow	2017-03-25 01:44:47	Fantastic	1490449522-5255-pip.jpg
5	5	4	LaLALA	Hehe	2017-03-25 01:45:25	Good	1490449559-1352-hurray.gif
6	5	5	Rated	NO	2017-03-25 01:46:02	Below Average	1490449604-7884-mae.jpg
7	5	1	Raziel is the best	Being modest! Am I?	2017-03-25 02:28:48	Average	1490449632-5901-toad.png

Figure 31 Event Review Data

Event Venue

```
SELECT * FROM `eventvenue`
```

id	event	venue	bookedBy	bookedOn
1	2	1	1	2017-03-03 12:24:41
2	1	1	2	2017-03-03 02:08:00
3	6	2	3	2017-03-13 08:49:02
5	6	1	1	2017-03-13 08:51:57
6	5	1	1	2017-03-03 12:55:06
7	11	2	1	2017-04-23 02:59:38

Figure 32 Event Venue Data

Gallery

```
SELECT * FROM `gallery`
```

id	event	note	img
1	1	Hallway! Gate.	1490449831-4787-Phoenix_Auditorium.jpg
2	1	It was awesome!!!	1490449879-7282-ev1.jpg
3	1	Croud was great!	1490449905-9638-ev2.jpg
4	1	A beautiful image of event!	1490449948-4604-ev3.jpg
5	3	Butter Butter Butter fly!	1490450723-9922-ev4.jpg
6	3	Autumn has come!	1490450778-5013-ev6.jpg
7	3	Autumn has come! A beautiful one!	1490450803-4557-ev7.jpg

Figure 33 Gallery data

Guest

```
SELECT * FROM `guest`
```

id	user	event	position	status
1	2	1	Guest of Honor	Attending
2	3	2	Guest of Honor	Attending
3	1	1	Admin	Attending
4	3	4	Member	Not Attending
5	4	3	Member	May Be
6	5	2	Guest	May Be
7	1	3	V.I.P	Not Attending
8	1	5	Guest of Honor	Attending
9	4	1	Member	Not Attending
10	6	1	Guest	May Be
11	5	1	V.I.P	May Be
12	3	3	Member	Not Attending
13	5	3	Member	May Be
14	7	3	Member	May Be
15	3	5	Admin	Attending
16	2	3	Guest of Honor	May Be
17	6	5	Guest	May Be
18	8	5	Guest	May Be
19	2	4	V.I.P	Attending
20	1	4	Member	Attending
21	2	2	Admin	Attending
22	3	1	Guest	May Be
23	3	6	Admin	May Be
24	1	7	Admin	Attending
25	1	8	Admin	Attending

Figure 34 Guest Data

Image Comment

```
SELECT * FROM `imagecomment`
```

id	image	user	comment	datetime
1	1	1	My Hallway is tremendously big.	2017-03-25 22:27:27
2	1	1	Ho HO HO!!	2017-03-25 23:00:27
3	1	2	The venue was quite nice... I just loved it. Looki...	2017-03-25 23:01:39
4	1	2	Can you tell when next event will be?	2017-03-25 23:04:24
5	1	3	Yes please!	2017-03-25 23:05:49
6	1	3	I am waiting...	2017-03-25 23:06:50

Figure 35 Image Comment Data

Invitation

```
SELECT * FROM `invitation`
```

id	event	user	message	date
1	1	2	You are requested to visit.	2017-03-11 12:13:21
2	2	3	Hello there!!!	2017-03-11 12:13:21
3	1	1	Please Come	2017-03-11 12:13:21
4	4	3	Coming right?	2017-03-11 12:13:21
5	3	4	Welcome!	2017-03-11 12:13:21
6	2	5	Ho HO HO OHOHOHOH	2017-03-11 12:13:21
7	3	1	Will you?	2017-03-11 12:13:21
8	5	1	Please visit and play with us.	2017-03-11 12:13:21
9	1	4	You are allowed to visit and complete the projects...	2017-03-11 12:13:21
10	1	6	You are allowed to visit and complete the projects...	2017-03-11 12:13:21
11	1	5	You are the legend!! TEKKEN 7	2017-03-11 12:13:21
12	3	3	This is a testing event to find if users visit and...	2017-03-11 12:13:21
13	3	5	This is a testing event to find if users visit and...	2017-03-11 12:13:21
14	3	7	This is a testing event to find if users visit and...	2017-03-11 12:13:21
15	5	3	You created the event!	2017-03-11 12:13:21
16	3	2	Welcome!!	2017-03-11 12:13:21
17	5	6	Please come to the event. We all are waiting for y...	2017-03-13 09:05:14
18	5	8	Please come to the event. We all are waiting for y...	2017-03-13 09:05:14
19	4	2	Will you come? Please??	2017-03-13 09:11:41
20	4	1	Chup chaap aa jaa!!!!	2017-03-13 09:12:33
21	2	2	You are the organizer!	2017-03-15 07:52:35
22	1	3	You are valuable to us!	2017-03-15 09:07:36
23	6	3	You organized the event!	2017-03-15 09:27:33
24	7	1	You created the event!	2017-03-31 08:23:53
25	8	1	You created the event!	2017-03-31 08:34:54

Figure 36 Invitation Data

Item

```
SELECT * FROM `item`
```

id	title	description	type	img
1	Pasta	Yummy Pasta for starters.	Chineese	1490450857-5255-pasta.jpg
2	Mc Aloo Tikki	A great meal.	Fast Food	1490450879-3780-mcall.jpg
3	Mc Wrap	Lorem ipsum dolor sit amet, consectetur adipisicin...	Fast Food	1490450910-7778-mcwra.jpg
4	Ramen	Pasta of china. ;)	Chineese	1490450943-8629-ramen.jpg
5	Donalad Duck	Quack Quack Quack, the signature.	Disney	1490451034-9249-don.jpg
6	Chip'n'Dale	Rescue Rangers	Disney	1490451066-4123-chip.jpg
7	Ariel	The little mermaid. Aquatic themed party elements.	Disney	1490451115-3259-ariel.jpg

Figure 37 Item Data

Item Review

```
SELECT * FROM `itemreview`
```

id	event	item	user	title	description	posted_on	rating	img
1	1	1	1	Good	It was really yummy!!	2017-03-25 02:12:04	Fantastic	1490451191-9840-lv.jpg
2	2	1	2	Not so good	I really can't say	2017-03-25 02:13:14	Average	1490451206-6669-1000.jpg
3	2	1	3	Not so good	It was okay	2017-03-25 07:13:14	Average	1490451206-6669-1000.jpg

Figure 38 Item Review

Schedule

```
SELECT * FROM `schedule`
```

id	event	title	description	datetime
1	1	Synopsis	Submitting the synopsis for project before 31st De...	2016-11-20 11:00:00
2	1	Project Design	Creating design for a project helps to know what t...	2016-11-25 08:15:00
3	3	Checks	Test test and TEST	2017-03-03 14:26:00
4	1	Another task	Create and edit and modify	2017-03-12 12:24:00
5	6	Induction	Welcome all the guests and see if they are up for ...	2017-10-03 12:00:00
6	6	Tally Guests	Hello Hello mic testing.	2017-03-10 14:36:00
7	11	Coding	Validate Coding	2017-04-29 12:15:00
8	11	Diagrams	Confirm report diagrams	2017-04-29 01:15:00

Figure 39 Schedule Data

Task

```
SELECT * FROM `task`
```

id	checklist	title	details	status	deadline	img	assigned_to
1	1	Bring Study Material	It is a must have.	Assigned	2017-03-25 01:23:45	1490451281-8060-ts1.jpg	1
2	4	New Created Task	Do it DO IT CSS	Working	2017-03-04 11:13:00	1490451290-4455-1489646774-8179-93 - Record Cleara...	2
3	1	Another task calculation	Do it complete and get rewarded.	Completed	2017-03-10 15:15:00	1490451436-4473-bv1.jpg	1
4	9	Go to bank	Go and find all the details for loan process.	Completed	2017-03-11 15:15:00	1490451451-9479-1489568976-6422-1553 - Lock.png	3
5	9	Find Manager	If he helps!	Completed	2017-03-11 15:40:00	1490451480-1828-per.jpg	3
6	7	A new task	Lorem ipsum dolor sit amet, consectetur adipiscing...	Assigned	2017-03-16 12:15:00	1490451507-3557-limie.jpg	1
7	7	Another task	Nam condimentum odio faucibus tellus vestibulum et...	Assigned	2017-03-16 13:15:00	1490451531-5185-c.jpg	2
8	1	New Task	A new task just to ensure the aside works.	Assigned	2017-03-31 23:59:00	1490451550-1641-te.jpg	1
9	11	Arrange Mics	You should find if they can be used.	Assigned	2017-04-10 00:00:00	1491024345-6189-microphone.png	1
10	23	Mics	Find a good mic provider	Assigned	2017-04-29 13:12:00	1492916357-3889-default-discount.png	1

Figure 40 Task Data

User

```
SELECT * FROM `user`
```

id	first_name	last_name	user_name	password	address	email	img	type
1	Lakshay	Verma	lakshay	8946553	Jalandhar Cantt	verma_lakshay@live.in	1490450106-6262-lk.jpg	admin
2	Mandeep	Kaur	mandeep	1234567	Jalandhar	mandeepshabina@gmail.com	1490448145-4361-1489396258-4222-Anime_Girl-wallpap...	admin
3	Raziel	Sarafan	Raziel	123456	Nosgoth	raziel@outlook.in	1490451605-7488-lok.png	admin
4	Dastan	Timeer	Dastan	123456	Persia	lakshayverma@outlook.in	1490448482-1468-1488528010-2486-cyanogenmod logo.p...	admin
5	Heihachi	Mishima	heihachi	tekken	Japan	heihachi@tekken.com	1490448615-9277-mario.jpg	admin
6	Jhon	Constantine	jhonny	lalala	New Orleans	jhonny.boy@constantine.us	1490450280-3102-dead.png	admin
7	Lufy	Monkey	hancock	123456	Grand Line	lufy@strawhats.jp	1490448660-4437-main-qimg-c0f2e7c8e5fb40c52acd389e...	admin
8	Boa	Hancock	hanboa	987654	Amazon	boa@op.com	1490448716-9016-boa.jpg	admin

Figure 41 User Data

Venue

```
SELECT * FROM `venue`
```

id	name	address	description	capacity	img
1	Vegas Paladium	House No 8 Mohalla No 14 Jalandhar Cantt	Lorem ipsum dolor sit amet, consectetur adipiscing...	85	1490448816-9151-vegas.jpg
2	Nevada Consertium	Mel ullum vulputate eu. Usu essent aperiam errorib...	Mea ad inani aeque interesset, pri mucius principe...	150	1490448869-5649-dinner.jpg
3	Next Gen Hall	Aliquam rhoncus magna nec elit placerat commodo. A...	Cum sociis natoque penatibus et magnis dis parturi...	150	1490448921-4705-room.jpg

Figure 42 Venue Data

Complete Project Coding

Client Pages

Index.php

```
<?php
$nav_only = TRUE;
$page_title = "Welcome";
include './layouts/header.php';
?>
<!-- Page Content -->
<div class="container-fluid">

    <!-- Heading Row -->
    <div class="row">
        <div class="col-md-8 site_logo" >
<!--            <img class="img-responsive img-rounded"
src="./dependencies/images/wall.jpg" alt="">-->
            <?php include './layouts/site_logo.php'; ?>
        </div>
        <!-- /.col-md-8 -->
        <div class="col-md-4">
            <h1><?php echo DEVELOPER_NAME; ?></h1>
            <p>
                I have created this web site for the
fulfillment of my <strong>Master's</strong> project.
                As per the guidelines a working project is
required in 6th semester of my degree.
                So here it is, <?php echo SITE_TITLE; ?>
my own personal project. Created from scratch.
                Made with <span class="glyphicon
glyphicon-heart"></span> in Jalandhar.
            </p>
            <?php if ($session->is_logged_in()) { ?>
                <a class="btn btn-primary btn-lg"
href="./view_event.php">Get Started</a>
            <?php } else { ?>
                <a class="btn btn-primary btn-lg"
href="./login.php">Get Started</a>
            <?php } ?>
        </div>
        <!-- /.col-md-4 -->
    </div>
    <!-- /.row -->

    <hr>

    <!-- Call to Action Well -->
    <div class="row">
        <div class="col-lg-12">
            <div class="well text-center">
                <?php echo SITE_MOTO; ?>
            </div>
```

```
            </div>
            <!-- /.col-lg-12 -->
        </div>
        <!-- /.row -->

        <!-- Content Row -->
        <div class="row">
            <?php include './layouts/site_branding.php'; ?>
        </div>
        <!-- /.row -->

        <!-- Footer -->

    </div>

    <?php include './layouts/footer.php'; ?>
```

About.php

```
    <?php
    $nav_only = FALSE;
    $page_title = "About Site";
    include './layouts/header.php';
    ?>

    <div class="container-fluid">
        <article id="site" class="panel panel-default">
            <h3 class="panel-heading">Site</h3>
            <div class="panel-body">
                <p>
                    Event management system is used to manage
    all the activity related to event. In any event many
    service providers work simultaneously and it is very hard
    to manage these providers. It is also important for event
    organizer that he has all the contacts details of these
    service providers so that he can contact them any time to
    plan an event at given time. To manage all these activity
    we have developed this software. To get success in the
    event management business, user should have strong network
    contacts of service provider. These contacts are
    essentially providers of specific services who can be
    mobilized quickly to participate in any given event. To
    make an event successful event manager needs different
    service provider like Sound systems services, Lighting
    providers, Canteen services, stage construction and so on.
    In present system Event Company have to do all management
    work manually. They keep all payment information on
    papers. There is no system to check the past expenses on
    any event. To do this they have to check payment register
    and this task is very time consuming and tiresome. Keeping
    all these problem in mind we have developed this system.
    This system helps the event management company to manage
```

```
their paper work online and they can also retrieve report
of last event they have completed.
            </p>

            <p>
                Event marketing is growing atâ€¯a rate of
three times that of traditional advertising. Though
relatively small compared to the major components of the
marketing communications mix-advertising, sales promotions
and P-O-P communications-expenditures on event sponsorship
are increasing.â€¯Corporate sponsorships inâ€¯India in
2001 were estimated at$3.9 billion-with 65% of this total
going toâ€¯sports events and most of the remainder spent
on sponsoring entertainment tours or festival and fairs.
Thousands of companies invest in some form of event
sponsorship. Defined, eventâ€¯marketing is a form of brand
promotion that ties a brand toâ€¯a meaningful athletic,
entertainment, cultural, social or other type ofâ€¯high-
interest public activity. Event marketing is distinct from
advertising, salesâ€¯promotion, point-of-purchase
merchandising, or public relations, butâ€¯it generally
incorporates elements from all of these promotional tools.
Event promotions have an opportunity to achieve success
because,â€¯unlike other forms of marketing communications,
events reach people when they are receptive toâ€¯marketing
messages and capture people in a relaxed atmosphere.
            </p>

            <p>
                Event marketing is growing rapidly
becauseâ€¯it provides companies alternatives to the
cluttered mass media, an ability to segment on a local or
regional basis, and opportunities for reaching narrow
lifestyle groups whose consumption behaviour can be linked
with the local event. MasterCard invested an estimated $25
millionâ€¯in sponsoring the nine-city World Cup soccer
championship in the United States in 1994 and will likely
sponsorâ€¯other big events in many countries as well.
            </p>
        </div>
    </article>

    <article id="developer" class="panel panel-default">
        <h3 class="panel-heading">Events Development</h3>
        <div class="panel-body">
            <p>
                These days, event management services are
in high demand, as more events are being arranged in
increasing numbers. Over the past ten to fifteen years,
the event management industry has witnessed huge growth.
Each year, across the world, nearly $500 billion is spent
on planned events.
                International Event management companies
are hired to organize a wide range of events. These
```

companies can organize events for small groups of people, or big events with thousands of attendees. Lots of businesses use event management services, to ensure that their events run smoothly and professionally. Event planning is a stressful, time consuming and expensive activity. An event management company has the expertise and industry contacts to offer a dependable service, at an affordable price. Letâ€™s take a closer look at how these companies approach organising different types of events:

<h4>Leisure Events </h4>

Leisure event management is an interesting and diverse field. It involves managing facilities like sports grounds, recreation centres, parks and entertainment venues. Also, it can involve managing a celebration or festival, sporting contest or concert.
The leisure events companies that manage these activities hire an engaged and enthusiastic workforce, who know their target audience. Moreover, these professionals have the expertise to carry out all the necessary administration and planning. The skill of the leisure event manager lies in his/her ability to understand what the event taking place means to those who are in attendance.
Also, a leisure event manager needs to incorporate value adding measures, to improve the experience of event goers. These measures could be delivered via technology, such as simulators or white knuckle rides in visitor attractions, or screen and sound systems in cinemas. The measures could be delivered through staff management, with service blueprinting to define staff roles in animating, motivating and engaging with customers. Or, the measures could be delivered through other work procedures and customer communication methods, to enhance the service quality, customer flow and circulation. This might encompass ticket purchasing, automatic entry, sight lines and signage, booking systems, and other similar technology that underpins the structure of an event.

<h4>Cultural Events </h4>

Often, cultural events are meant to enrich the cultural standing of the city in which they are held. Cultural events teams create unique festivals, memorable outdoor spectacles, accessible entertainment and unexpected arts. Often, these teams work alongside local government authorities. Every year, they will plan innovative cultural programmes and events, and advertise and oversee them from start to finish.
Managing a cultural event is a major undertaking. Festivals seem glamorous and exciting but in

reality, they are hard work in every respect. Cultural event managers have to deliver an artistic and cultural programme, which will appeal to the public. However, they have to take local sensitivities into account, adhere to strict planning and health and safety rules, and still make a profit.

Cultural event managers need skills in fundraising and finance, public relations and arts marketing. Typically, these events management companies have tight budgets, and are dependent on part time staff, enthusiastic volunteers and freelance contractors. Obviously, styles of cultural events management vary, based on the genre and size of events. Nonetheless, all cultural events are linked to the well being of the community and issues of identity. Also, they all tap into the visitor and tourist economy.

<h4>Personal Events </h4>

Invariably, it takes a lot of time, effort and dedication to organize a personal event. With catering arrangements, types of menus, guest lists, booking the venue and designing the dÃ©cor, there are countless details that have to be scheduled and coordinated. In the past, personal events management companies only dealt with indoor events. However, nowadays, these companies manage outdoor events as well.

A personal event management company will bring a party to life, with colors, patterns, spectacular backdrops and lighting effects that will enchant all the guests present. Indeed, these companies specialise in organising funky, uniquely themed events. The company will consult with the customer to discuss the precise requirements, and then tailor the event in line with the customerâ€™s budget. Venue booking, marquee erecting, menu planning and live entertainment provision is all included in the service.

Not all customers for these types of events require complete event management. Hence, personal event management services can be enlisted to organize only part of an event â€” for example, the hiring of entertainment or the catering. Initial consultations will determine which parts of the event would be best left to a professional organizer, and which parts the event hosts can tackle themselves. Often, the division of labour is decided by budget, and by the customerâ€™s relative expertise.

<h4>Organizational Events </h4>

Organizational events can include political, charitable and commercial events, as well as sales events, such as product launches, etc. A company that hosts an all day event for several thousand people

will require catering, entertainment and accommodation arranging for all the guests. Event staff will need to be recruited, a room to host the event will have to be chosen, seating arrangements will need to be determined and obviously, an event budget has to be established. An organisational events management company will have expertise in all of these areas.

Also, these companies might be hired to do more than simply plan and execute an event. They may be enlisted to implement a communications, marketing and brand building strategy as well. An organisational events manager understands the technical, logistical and creative factors that make an event successful. This includes audio visual production, event design, logistics, scriptwriting, negotiation, budgeting and of course, customer service. For bigger organisational events, in the preparation stage, the events manager has to make important decisions and choices linked to the eventâ€™s creative concept and design. To ensure that the event runs to plan, the events manager requires extensive technical design expertise, and a clear grasp of how to convey a companyâ€™s message in the public arena.

Obviously, event management is a multifaceted profession. The scale of the task involved in managing the above types of events is significant. It is hardly surprising that the industry is thriving.

```
            </p>
        </div>
    </article>

    <article id="tech" class="panel panel-info">
        <h3 class="panel-heading">Summary</h3>
        <div class="panel-body">
            <p>
```

Event management means any service provided in relation toâ€¯planning, promotion, organizing orâ€¯presentation of any arts, entertainment, business, sports or any other event andâ€¯includes any consultation provided in this regard The current condition of the event management companies in the city is not very well they are waiting for the response of the people so that in future they can encash it. In fact right now the role of event management companies is being played by either the hotels orâ€¯the amusement parks or theâ€¯community centres. In many cases newspapers work in collaboration with private parties to organize some event. The findings of the project clearly states that the future of event management is veryâ€¯bright in the city although presently itsâ€¯in nascent stage but the people of Delhi have started accepting such events in theâ€¯main line. Some such events are like theme based marriages and â€œLadies Sangeetâ€•â€¯in the marriages. Along with this the growing gathering in the fashion shows and musical concertsâ€¯is also remarkable. Right now the condition of event

```
management in Delhi is not very good. The response
ofâ€¯â€¯people has not been that good that it can attract
any event management company here as a full fledged
service provider, it will take some time to pick up
nicely. Currently the main target segment of these partial
or complete service providers are the defence class people
or the high class people have accepted these concepts much
widely then any otherâ€¯community. It is expected that
eventâ€¯management industry will gear up in Delhi in
nearâ€¯future. The concept needs to be popularized, by
proper media coverage. The existing companies should mould
their service according to the taste and preference of the
people. The scope of event management companies in Delhi
is notâ€¯very good. As per our findings and the
calculations the results are notâ€¯very encouraging at
present. But the projections help us to draw a conclusion
that there is a scope for eventâ€¯management companies in
near future.
            </p>
        </div>
    </article>
</div>

<?php include './layouts/footer.php'; ?>
```

New_Event.php

```
<?php
$edit_records = FALSE;
$nav_only = TRUE;
$page_title = "Create A new Event";
include './layouts/header.php';
members_only();
$current_event = new Event();
$current_event->id = 0;
?>
<div class="container-fluid">
    <div class="panel panel-primary col-md-8 col-md-
offset-2">
        <header class="panel-heading">
            <h3>
                <?= $page_title; ?>
            </h3>
        </header>
        <div class="panel-body">
            <?php $event = new Event(); ?>
            <form id="form" class="panel-body"
method="post" action="./logic/create_event.php"
enctype="multipart/form-data">
                <div class="row">
                    <legend>Basic Information</legend>
                    <div class="form-group col-md-5">
```

```
                        <label class="col-form-label"
for="name">Event name</label>
                        <input tabindex="1" id="name"
name="name" class="form-control" type="text" value="<?php
echo $event->name; ?>" required/>
                    </div>
                    <div class="form-group col-md-4">
                        <label class="col-form-label"
for="datetime">Date and Time</label>
                        <input tabindex="2" id="datetime"
name="datetime" class="form-control"  type="datetime-
local" value="<?php echo str_replace(" ", "T", $event-
>datetime); ?>"  required/>
                    </div>
                    <div class="form-group col-md-5">
                        <label class="col-form-label"
for="img">Image</label>
                        <input tabindex="4" id="img"
name="img" class="form-control" type="file"
accept="image/*" <?php echo ($event->img) ? '' :
'required'; ?>/>
                    </div>
                    <div class="form-group col-md-12">
                        <label class="col-form-label"
for="description">Event Description</label>
                        <textarea tabindex="5"
id="description" name="description" class="form-control"
required><?php echo $event->description; ?></textarea>
                    </div>
                </div>
                <div class="row">
                    <div class="form-group col-md-6">
                        <legend class="col-form-label"
for="item">Items</legend>
                        <select tabindex="6" id="item"
name="item[]" class="form-control" multiple>
                            <?php $groups =
Item::find_ordered(); ?>
                            <?php while ($group =
current($groups)): ?>

                                <optgroup label="<?=
ucwords(key($groups)) ?>">
                                    <?php while ($option =
current($group)): ?>
                                        <option value="<?=
$option["id"]; ?>" ><?= $option["title"]; ?></option>
                                        <?php
                                        next($group);
                                    endwhile;
                                    ?>
                                </optgroup>

                                <?php
```

```
                                next($groups);
                            endwhile;
                            ?>
                        </select>
                    </div>
                    <div class="form-group col-md-6">
                        <legend>Pre Created
Checklists</legend>
                        <?php
                        $checklist_options =
Created_Checklists::available_checklists();

                        while ($list_option =
current($checklist_options)) {
                            ?>
                            <div class="checkbox">
                                <label><input
type="checkbox" name="<?= $list_option; ?>"> <?=
ucwords($list_option); ?>(s) Management List</label>
                            </div>

                            <?php
                            next($checklist_options);
                        }
                        ?>


                        <!--<label class = "col-form-
label" for = "checklists">Pre created checklists</label>
                        <select tabindex = "7" id =
"checklists" name = "checklists" class = "form-control"
multiple>
                        <?php
//                            $options =
CheckList::find_precreated();
//                            include
'./layouts/data/options_list.php';
                        ?>
                        </select>                -->
                    </div>
                </div>

                <div class="row ">
                    <input id="organiser" name="organiser"
type="hidden" value="<?php echo $current_user->id; ?>">
                    <input id="table_name"
name="table_name" type="hidden" value="event"/>
                    <input tabindex="8" class="form-
control btn btn-primary" type="submit" value="Submit"/>
                    <input tabindex="9" class="form-
control btn " type="reset" value="Clear"/>
                </div>

            </form>
```

```
                <script>
                    var formRules = {
                        rules: {
                        },
                        messages: {
                        }
                    };
                </script>
            </div>

        </div>
    </div>
    </div>

    <?php include './layouts/footer.php' ?>
```

View_Event.php

```
    <?php

    $edit_records = FALSE;
    $page_title = "Manage Events";
    include './layouts/header.php';
    members_only();
    $events = Event::find_for_user($current_user->id);
    $invited_events = Invitation::find_past($current_user-
    >id);
    $upcoming_events =
    Invitation::find_upcoming($current_user->id);
    $current_event = (isset($_GET['event'])) ? $_GET['event']
    : 0;
    $current_event = Event::find_by_id($current_event);
    if (!$current_event) {
        if ($events) {
            $current_event = current($events);
        } else {
            $current_event = null;
        }
    }
    include './layouts/data/event_renderer.php';
    ?>
    <?php include './layouts/footer.php' ?>
```

View_Checklist.php

```
<?php
    $custom_header = TRUE;
    $custom_footer = TRUE;
    $edit_records = FALSE;
    $page_title = "Checklist";
```

```
include './layouts/header.php';
$current_list = (isset($_GET['list'])) ? $_GET['list'] :
0;
$current_list = CheckList::find_by_id($current_list);
$current_event = $current_list->get_event();
$m_pos = Guest::is_guest($current_user->id,
$current_event->id);
$m_pos = array_shift($m_pos);
$position = strtolower($m_pos->position);

if ($position == 'admin' || $position == 'member' ||
$current_event->organiser == $current_user->id) {
    include './layouts/data/checklist_renderer.php';
} else {
    ?>
    <div class="container-fluid">
        <div class="panel panel-danger">
            <h1 class="panel-heading">
                Restricted!!
            </h1>
            <p class="panel-body">
                "You are a <strong class="text-
capitalize"><?php echo $position; ?> </strong> and only
Organizer or Members of event are allowed to see
checklists."
            </p>
        </div>
    </div>
    <?php
}
include './layouts/footer.php'
?>
```

View_Image.php

```
<?php
$nav_only = TRUE;
$page_title = "Images";
include './layouts/header.php';
members_only();
if (isset($_GET['image_id'])) {
    $id = $_GET['image_id'];
    $image = Gallery::find_by_id($id);
    if ($image->event) {
        $image_found = TRUE;
        if (Guest::is_guest($current_user->id, $image-
>event)) {

            $can_comment = TRUE;
            $comments = $image->get_comments();
        } else {
            $can_comment = FALSE;
            $msg = "You can not comment on the image.";
```

```
        }
    } else {
        $image_found = FALSE;
        $msg = "Image Not Found...";
    }
}
?>
<!-- Page Content -->
<article class="container-fluid">
    <?php if ($image_found): ?>
        <div class="col-md-9">
            <section id="image_viewer">
                <img id="photograph" src="<?php echo
$image->image_source(); ?>"/>
                <blockquote class="caption">
                    <?php echo $image->note; ?>
                </blockquote>
            </section>
        </div>
        <?php if ($can_comment): ?>
            <aside id="img_comments" class="col-md-3 panel
panel-primary">
                <h2 class="panel-heading">Comments</h2>
                <section class="panel-body">
                    <ul class="list-group">
                        <?php
                        while ($comment =
current($comments)):
                            ?>
                            <li class="list-group-item
row">
                                <div class="col-md-4">
                                    <?php echo $comment-
>get_user()->avatar("64px", "img img-thumbnail zoom-img
stay","org"); ?>
                                </div>
                                <div class="col-md-8">
                                    <?php echo $comment-
>comment; ?>
                                </div>
                            </li>
                            <?php
                            next($comments);
                        endwhile;
                        ?>
                    </ul>
                </section>
                <form class="form-inline" method="post"
action="./logic/comment.php">
                    <input name="comment" type="text"
class="form-control">
                    <input name="image" type="hidden"
value="<?php echo $image->id; ?>"/>
```

```
                    <input name="redirect_url"
type="hidden" value="<?php echo $_SERVER["REQUEST_URI"];
?>"/>
                    <button type="submit" class="btn btn-
primary btn-sm">
                        <span class="glyphicon glyphicon-
chat"></span>
                        Comment
                    </button>
                </form>
            </aside>
        <?php endif; ?>
    <?php endif; ?>
</article>
<?php include './layouts/footer.php'; ?>
```

Backend Logic

Config.php

```
<?php
// Database Constants
defined('DB_SERVER')    ? null : define("DB_SERVER",
"localhost");
defined('DB_NAME')      ? null : define("DB_NAME",
"events_db");
defined('DB_USER')      ? null : define("DB_USER",
"root");
defined('DB_PASS')      ? null : define("DB_PASS", "");
?>
```

```
<?php
// Site Details
defined('SITE_TITLE') ? null : define('SITE_TITLE',
"Lakshay");
defined('SITE_MOTO') ? null : define('SITE_MOTO', "Light
as air, we create and manage events seamlessly.");
defined('DEVELOPER_NAME') ? null :
define('DEVELOPER_NAME', "Lakshay Verma");
defined('DEVELOPER_INFO') ? null :
define('DEVELOPER_INFO', "146433412");
defined('DEVELOPER_MAIL') ? null :
define('DEVELOPER_MAIL', "verma_lakshay@live.in");
?>
```

Database.php

```
<?php

require_once(LIB_PATH . DS . "config.php");

class MySQLDatabase {
```

```
    private $connection;

    public function __construct() {
        $this->open_connection();
    }

    // Connection functions
    public function open_connection() {
        $this->connection = mysqli_connect(DB_SERVER,
DB_USER, DB_PASS, DB_NAME);
        if (!$this->connection) {
            die("Database connection failed: " .
                    mysqli_connect_error() .
                    " (" . mysqli_connect_errno() . ")"
            );
        }
    }

    public function close_connection() {
        if (isset($this->connection)) {
            mysqli_close($this->connection);
            unset($this->connection);
        }
    }

    // Database query functions
    public function query($sql) {
        $result = mysqli_query($this->connection, $sql);
        $this->confirm_query($result, $sql);
        return $result;
    }

    private function confirm_query($result, $sql = "") {
        if (!$result) {
            die("<p>Database query <big>{$sql}</big>
failed.</p>");
        }
    }

    public function escape_value($string) {
        $escaped_string = mysqli_real_escape_string($this-
>connection, $string);
        return $escaped_string;
    }

    // Database neutral functions
    public function fetch_array($result_set) {
        return mysqli_fetch_array($result_set);
    }

    public function num_rows($result_set) {
        return mysqli_num_rows($result_set);
    }
```

```
    public function insert_id() {
        // get the last id inserted over the current
database connection
        return mysqli_insert_id($this->connection);
    }

    public function affected_rows() {
        return mysqli_affected_rows($this->connection);
    }

}

$database = new MySQLDatabase();
$db = & $database;
?>
```

Database_Object.php

```
<?php

//If it is going to need database, then it's probably
smart to require it before we start
require_once(LIB_PATH . DS . 'database.php');

class DatabaseObject {

    public static function find_by_sql($sql = "") {
        global $database;
        $result_set = $database->query($sql);
        $object_array = array();                // NOTE
solution to problem #1
        while ($row = $database->fetch_array($result_set))
{
            $object_array[] = static::instantiate($row);
// NOTE using static:: instead of self:: for late static
binding.
        }
        return $object_array;
    }

    public static function find_all() {
        return self::find_by_sql("SELECT * FROM " .
static::$table_name);
    }

    public static function find_limited($limit = 10, $page
= 1) {
        $offset = ($page - 1) * $limit;
        return self::find_by_sql("SELECT * FROM " .
static::$table_name . " limit $limit offset $offset");
    }

    public static function find_by_id($id) {
        if (empty($id)) {
```

```
            return false;
        }
        global $database;
        $id = $database->escape_value($id);
        $result_array = self::find_by_sql("SELECT * FROM "
. static::$table_name . " WHERE id = {$id} LIMIT 1");
        return !empty($result_array) ?
array_shift($result_array) : false;
    }

    public static function count_all() {
        global $database;
        $sql = "SELECT COUNT(*) FROM " .
static::$table_name;
        $result_set = $database->query($sql);
        $row = $database->fetch_array($result_set);
        return array_shift($row);
    }

    protected static function instantiate($record) {
        $object = new static;
        foreach ($record as $attribute => $value) {
            if ($object->has_attribute($attribute)) {
                $object->$attribute = $value;
            }
        }
        return $object;
    }

    private function has_attribute($attribute) {
        $object_vars = $this->attributes();
        return array_key_exists($attribute, $object_vars);
// all we need to confirm is the key exists irrespective
of the value.
    }

    public function attributes() {
        $attributes = array();
        foreach (static::$db_fields as $field) {
            if (property_exists($this, $field)) {
                $attributes[$field] = $this->$field;
            }
        }
        return $attributes;
    }

    protected function sanitized_attributes() {
        global $database;
        $clean_attributes = array();
        // sanitize the values before submitting
        // NOTE: does not alter the actual value of each
attribute

        foreach ($this->attributes() as $key => $value) {
```

```
            $clean_attributes[$key] = $database-
>escape_value($value);
        }
        return $clean_attributes;
    }

    function insertion_attributes() {
        global $database;
        $clean_attributes = array();
        // sanitize the values before submitting
        // NOTE: does not alter the actual value of each
attribute

        foreach ($this->attributes() as $key => $value) {
            if ($key != "id") {
                $clean_attributes[$key] = $database-
>escape_value($value);
            }
        }
        return $clean_attributes;
    }

    public function validate_attributes($attributes =
array()) {
        foreach ($attributes as $attribute) {
            if ($this->has_attribute($attribute) &&
empty($this->$attribute)) {
                return false;
            }
        }
        return true;
    }

    public function show_attributes($attributes = array())
{
        $string = "<ul>";
        foreach ($attributes as $attribute) {
            $string .= "<strong>{$attribute}</strong>
{$this->$attribute}";
            if ($this->has_attribute($attribute) &&
empty($this->$attribute)) {
                $string .= "<strong>{$attribute}</strong>
{$this->$attribute}";
            }
        }
        $string .= "</ul>";
        return $string;
    }

    public function get_db_fields() {
        return static::$db_fields;
    }

    // CRUD Functions
```

```
    public function save() {
        // A new record won't have an id yet.
        $this->upload_dir();
        return (isset($this->id) || ($this->id != null)) ?
$this->update() : $this->create();
    }

    public function upload_dir() {
        $upload_dir = UPLOAD_PATH . DS .
static::$table_name;
        if (!is_dir($upload_dir)) {
            mkdir($upload_dir, 0777, true);
        }
        return $upload_dir;
    }

    public static function image_dir() {
        return 'uploads' . DS . static::$table_name;
    }

    public function upload_img($img) {
        $tmp_file = $img['tmp_name'];
        $name = basename($img['name']);
        $destination = time() . "-" . rand(1000, 9999) .
"-" . $name;
        move_uploaded_file($tmp_file, $this->upload_dir()
. DS . $destination);
        return $destination;
    }

    public function create() {
        global $database;
        $attributes = $this->insertion_attributes();
        $sql = "INSERT INTO " . static::$table_name . "
(";
        $sql .= join(", ", array_keys($attributes));
        $sql .= ") VALUES (\"";
        $sql .= join("\", \"", array_values($attributes));
        $sql .= "\")";

        if ($database->query($sql)) {
            $this->id = $database->insert_id();
            return true;
        } else {
            return false;
        }
    }

    public function init_members() {
        // DO NOTHING;
    }

    public function table_edit($editable = FALSE) {
        global $current_user;
```

```
        if (method_exists($this, "init_members")) {
            $this->init_members();
        }

        if ($current_user->is_admin() && $editable) {
            return "<td id=\"record-{$this->id}\"
class=\"col-sm-12 col-md-2\">"
                    . "<form method=\"post\"
action=\"./tableForms/delete.php\" class=\"col-md-6\">"
                    . "<button type=\"submit\" class=\"btn
btn-small btn-danger\">"
                    . "<span class=\"glyphicon glyphicon-
trash\"></span>"
                    . "</button>"
                    . "<input type=\"hidden\"
name=\"table_name\" value=\"" . static::$table_name .
"\"/>"
                    . "<input type=\"hidden\"
name=\"id\"value=\"" . $this->id . "\"/>"
                    . "<input type=\"hidden\"
name=\"redirect_url\"value=\"" . $_SERVER["REQUEST_URI"] .
"\"/>"
                    . "</form>"
                    . "<div class=\"col-md-6\">"
                    . "<a class=\"btn btn-warning\"
href=\"?table=" . static::$table_name . "&id=$this->id\">"
                    . "<span class=\"glyphicon glyphicon-
edit\"></span>" . $this->id
                    . "</a>"
                    . "</div>"
                    . "</td>";
        } else {
            return "<td class=\"col-sm-12 col-md-2\">" .
$this->id . "</td>";
        }
    }

    public function update() {
        global $database;
        $attributes = $this->sanitized_attributes();
        $attribute_pairs = array();
        foreach ($attributes as $key => $value) {
            $attribute_pairs[] = "{$key}='{$value}'";
        }
        $sql = "UPDATE " . static::$table_name . " SET ";
        $sql .= join(", ", $attribute_pairs);
        $sql .= " WHERE id=" . $database-
>escape_value($this->id);
        $database->query($sql);
        return ($database->affected_rows() == 1) ? true :
false;
    }
```

```
    public function delete() {
        global $database;
        $sql = "DELETE FROM " . static::$table_name . " ";
        $sql .= "WHERE id=" . $database-
>escape_value($this->id) . " ";
        $sql .= "LIMIT 1";
        $database->query($sql);
        return ($database->affected_rows() == 1) ? true :
false;
    }

    public function datetime($format = "h:i a, F d Y") {
        if ($this->datetime) {
            return static::format_datetime($format, $this-
>datetime);
        }
    }

    public static function format_datetime($format = "h:i
a, F d Y", $datetime) {
        $dt = new DateTime($datetime);
        return $dt->format($format);
    }

    public static function form_date($date) {
        return static::format_datetime("Y-m-d", $date) .
'T' . static::format_datetime("h:i:s", $date);
    }

    public function img() {
        return $this->image_dir() . DS . $this->img;
    }

    public function image_source() {
        return $this->image_dir() . DS . $this->img;
    }

    public function image() {
        return $this->avatar();
    }

    public function avatar($image_size = "72px", $class =
"img img-thumbnail", $tag_title = "-") {
        $title = ($tag_title == "-") ? "" : $this->name();
        $class = $class . " img-square img-" .
str_replace("px", "", $image_size);
        if ($this->has_attribute('img')) {
            $startTag = "<img";
            $endTag = "/>";
            $content = " width=\"$image_size\""
                    . " height=\"$image_size\""
                    . " alt=\" " . $this->name() . "\""
                    . " title=\" " . $title . "\""
                    . " class=\"$class\""
```

```
                        . " src=\"" . $this->img() . "\"";
            } else {
                $startTag = "<span>";
                $endTag = "</span>";
                $content = $this->name();
            }

            return $startTag
                    . $content
                    . $endTag;
        }

        public function intro($image_size = "72px", $class =
    "", $classImg = "img img-thumbnail", $title = "org") {
            global $session;
            $user = $session->get_user_object();
            if ($user->is_admin()) {
                return "<a href=\"./list_tables.php?table=" .
    static::$table_name . "&id=$this->id\" class=\"$class\" >"
                        .
                        $this->avatar($image_size, $classImg,
    $title)
                        . "</a>";
            } else {
                return $this->avatar($image_size, $classImg,
    $title);
            }
        }

        public function title() {
            return $this->name();
        }

    }

    ?>
```

Functions.php

```
    <?php

    // Required functions
    function __autoload($class_name) {
        $class_name = strtolower($class_name);
        $path = LIB_PATH . DS . "{$class_name}.php";
        if (file_exists($path)) {
            require_once($path);
        } else {
            die("The file {$class_name}.php could not be
    found.");
        }
    }

    function redirect_to($location = NULL) {
```

```
    if ($location != NULL) {
        if (empty($location)) {
            $location = "index.php";
        }
        if (!headers_sent()) {
            header("Location: {$location}");
        }
        exit;
    }
}

// HTML Functions
function include_layout_template($template = "") {
    include(SITE_ROOT . DS . 'layouts' . DS . $template);
}

function output_message($message = "", $class = "") {
    if (!empty($message)) {
        return "<p class=\"message
{$class}\">{$message}</p>";
    } else {
        return "";
    }
}

function strip_zeros_from_date($marked_string = "") {
    // remove marked zeros
    $no_zeros = str_replace("*0", "", $marked_string);

    // remove remaining marks
    $cleaned_string = str_replace("*", "", $no_zeros);
    return $cleaned_string;
}

// LOG functions
function log_action($action, $message = "") {
    $file = SITE_ROOT . DS . 'logs' . DS .
'site_logs.txt';
    if ($handle = fopen($file, 'a')) {
        $log = "";
        $log .= strftime("%Y-%m-%d %H:%M:%S", time()) . "
| ";
        $log .= $action . " " . $message . "\r\n";
        fwrite($handle, $log);
        fclose($handle);
    }
}

function get_all_logs() {
    $file = SITE_ROOT . DS . 'logs' . DS .
'site_logs.txt';
    $log = "";
    if ($handle = fopen($file, 'r')) {
        $log = fread($handle, filesize($file));
```

```
        fclose($handle);
    }
    $tempLog = trim($log);
    if (empty($tempLog)) {
        $log = "No log information available yet.";
    }
    return $log;
}

function wipe_all_logs($user_id) {
    $file = SITE_ROOT . DS . 'logs' . DS .
'site_logs.txt';

    $username = User::find_by_id($user_id)->username;
    if ($handle = fopen($file, 'w')) {
        $log = "user: {$username} with id: " . $user_id .
" cleared all previous logs on ";
        $log .= strftime("%Y-%m-%d %H:%M:%S", time());
        $log .= "\r\n";
        fwrite($handle, $log);
        fclose($handle);
    }
}

function admins_only() {
    give_access(TRUE);
}

function inside_persons_only() {
    give_access(FALSE, TRUE);
}

function members_only() {
    give_access(FALSE, FALSE);
}

function give_access($onlyAdmin = true, $onlyUsers =
false) {
    global $session;
    $session->request_uri($_SERVER['REQUEST_URI']);
    if ($onlyUsers && $session->get_user_object()->type ==
'client') {
        $session->message("Restricted Access.");
        redirect_to('index.php');
    }
    if ($onlyAdmin) {
        if (!$session->is_admin()) {
            $session->message("Only admins can view the
requested page.");
            redirect_to('index.php');
        }
        // else keep going
    } else {
        if (!$session->is_logged_in()) {
```

```
            $session->message("You need to Login/Register
first.");
            redirect_to('login.php');
        }
    }
}

function get_all_tables() {
    global $database;
    $result_set = $database->query("show tables");
    $database_tables = array();
    while ($table_detail =
mysqli_fetch_array($result_set)) {
        $database_tables[$table_detail[0]] =
get_table_details($table_detail[0]);
    }
    return $database_tables;
}

function get_table_details($table_name) {
    global $database;
    $table_result = $database->query("show columns from
{$table_name}");
    while ($col = mysqli_fetch_assoc($table_result)) {
        $column[$col["Field"]] = $col["Type"];
    }
    return $column;
}

function get_input_type($column) {
    $col = strtoupper(substr($column, 0, 3));
    switch ($col) {
        case 'INT':
            return "number";
        case 'VAR':
            return 'text';
        case 'ENU':
            return 'select';

        case 'DAT':
        case 'TIM':
            return "datetime-local";

        default :
            return 'area';
    }
}

function create_input_element($name, $type, $default,
$classes) {
    $column = get_input_type($type);
    if ($column === 'area') {
```

```
        $string = "<textarea id=\"{$name}\"
name=\"{$name}\"
class=\"{$classes}\">{$default}</textarea>";
    } elseif ($column === 'select') {
        $optionsString = substr($type, strpos($type, '('),
strpos($type, ')'));
        $optionsString = str_replace('(', '',
$optionsString);
        $optionsString = str_replace(')', '',
$optionsString);
        $values = explode(',', $optionsString);

        $string = "<select id=\"{$name}\" name=\"{$name}\"
class=\"{$classes}\">";
        foreach ($values as $option) {
            $option = str_replace('\'', '', $option);
            $string .= "<option
value=\"{$option}\">{$option}</option>";
        }
        $string .= "</select>";
    } else {
        $string = "<input id=\"{$name}\" name=\"{$name}\"
class=\"{$classes}\" type=\"{$column}\" ";
        if ($name == "id") {
            $string .= " disabled ";
        } else {
            $string .= " value=\"{$default}\" ";
        }
        $string .= "/>";
    }
    return $string;
}

function get_checkbox_value($checkbox) {
    if (strtolower($checkbox) === 'on') {
        return TRUE;
    } else {
        return FALSE;
    }
}

function form_date_time(){
    return date("Y-m-d") . 'T' . date("h:i:s");
}
?>
```

Checklist.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
```

```
require_once(LIB_PATH . DS . "database.php");

class CheckList extends DatabaseObject {

    protected static $table_name = "Checklist";
    protected static $db_fields = array('id', 'title',
'user', 'event', 'created_on', 'img');
    public $id;
    public $title;
    public $user;
    public $event;
    public $created_on;
    public $img;
    public $userObj, $eventObj;

    public static function make($title, $user, $event,
$created_on, $img) {
        $checklist = new self;
        $checklist->title = $title;
        $checklist->user = $user;
        $checklist->event = $event;
        $checklist->created_on = $created_on;
        $checklist->img = $img;

        return $checklist;
    }

    public function name() {
        return $this->title;
    }

    public function init_members() {
        if (!$this->userObj) {

            $this->userObj = User::find_by_id($this-
>user);
        }

        if (!$this->eventObj) {
            $this->eventObj = Event::find_by_id($this-
>event);
        }
    }

    public function anchor($image_size = "72px", $class =
"", $classImg = "img img-thumbnail", $title = "org") {

        $anchor = "<a
onclick=\"showPopup(this.href);return(false);\"
href=\"./view_checklist.php?list={$this->id}\""
                . "class=\"$class\""
                . ">"
                . $this->avatar($image_size, $classImg,
$title)
```

```
                . "<em>"
                . $this->name()
                . "</em>"
                . "</a>";

        return $anchor;
    }

    public function get_author() {
        $this->init_members();
        return $this->userObj;
    }

    public function get_event() {
        $this->init_members();
        return $this->eventObj;
    }

    public static function find_all_for_event($event) {
        $sql = "select * from "
                . static::$table_name
                . " where event=$event";
        return self::find_by_sql($sql);
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Checklist Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        User
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Created On
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit) {
        return "
            <tr class=\"row\">"
```

```
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_author()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_event()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->created_on . "</td>"
                . "</tr>";
    }

}

?>
```

Created_Checklists.php

```
<?php

require_once(LIB_PATH . DS . "checklist.php");

class Created_Checklists {

    const AGENDA = 'agenda';
    const ARTIST = 'artist';
    const DECORATIONS = 'decorations';
    const GUEST = 'guest';
    const LEGAL = 'legal';
    const LIGHTS = 'light';
    const SOUND = 'sound';
    const VISUAL = 'visual';

    public static function available_checklists() {
        $array = array(self::AGENDA, self::ARTIST,
self::DECORATIONS, self::GUEST, self::LEGAL, self::LIGHTS,
self::SOUND, self::VISUAL);
        return $array;
    }

    public static function checklist_primary($event,
$sound = TRUE, $lights = TRUE, $guest = TRUE, $visuals =
TRUE, $decorations = TRUE, $artist = TRUE) {
        global $session;
        $current_user = $session->get_user_object();
        $lists = array();
        if ($sound) {
            $lists[] = CheckList::make("Sound",
$current_user->id, $event->id, date("Y-m-d H:i:s"),
"defaults" . DS . "microphone.png");
        }
        if ($lights) {
```

```
            $lists[] = CheckList::make("Lights and
        Electricity", $current_user->id, $event->id, date("Y-m-d
        H:i:s"), "defaults" . DS . "light-bulb.png");
        }

        if ($guest) {
            $lists[] = CheckList::make("Guest Management",
        $current_user->id, $event->id, date("Y-m-d H:i:s"),
        "defaults" . DS . "guests.png");
        }
        if ($visuals) {
            $lists[] = CheckList::make("Party Visuals",
        $current_user->id, $event->id, date("Y-m-d H:i:s"),
        "defaults" . DS . "confetti.png");
        }
        if ($decorations) {
            $lists[] = CheckList::make("Party
        Decorations", $current_user->id, $event->id, date("Y-m-d
        H:i:s"), "defaults" . DS . "party.png");
        }
        if ($artist) {
            $lists[] = CheckList::make("Artist",
        $current_user->id, $event->id, date("Y-m-d H:i:s"),
        "defaults" . DS . "singer.png");
        }
        return $lists;
    }

    public static function
        checklist_organisational($event, $legal = TRUE, $agenda =
        TRUE, $sound = TRUE, $lights = TRUE, $guest = TRUE,
        $visuals = TRUE, $decorations = TRUE, $artist = TRUE) {
        global $session;
        $current_user = $session->get_user_object();
        $lists = static::checklist_primary($event, $sound,
        $lights, $guest, $visuals, $decorations, $artist);
        if ($legal) {
            $lists[] = CheckList::make("Legal Docs",
        $current_user->id, $event->id, date("Y-m-d H:i:s"),
        "defaults" . DS . "contract.png");
        }
        if ($agenda) {
            $lists[] = CheckList::make("Agenda",
        $current_user->id, $event->id, date("Y-m-d H:i:s"),
        "defaults" . DS . "notebook.png");
        }
        return $lists;
    }

}
```

Event.php

```
<?php
```

```
// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class Event extends DatabaseObject {

    protected static $table_name = "event";
    protected static $db_fields = array('id', 'name',
'description', 'organiser', 'datetime', 'img');
    public $id;
    public $name;
    public $description;
    public $organiser;
    public $img;
    public $datetime;
    public $items;

    public static function make($name, $description,
$organiser, $datetime) {
        $event = new self;
        $event->name = $name;
        $event->description = $description;
        $event->organiser = $organiser;
        $event->datetime = $datetime;

        return $event;
    }

    public function can_be_rated() {
        $dateFormat = 'Y-m-d';
// Year-Month-Date.                          (2017-03-17)
        $timeFormat = 'h:i:s a';
// Hours:Minutes:Seconds.                    (07:41:07 pm)
        $format = $dateFormat;          // Year-Month-Date
Hours:Minutes:Seconds.    (2017-03-17 07:41:07 pm)

        $selected_zone = new DateTimeZone('Asia/Kolkata');

        $date2 = new DateTime('now', $selected_zone);
        $date = new DateTime($this->datetime,
$selected_zone);

        $datediff = strtotime($date->format($format)) -
strtotime($date2->format($format));

        $daysBetween = floor($datediff / (60 * 60 * 24));

        if ($daysBetween < 0) {
            return TRUE;
        } else {
            return FALSE;
        }
    }
```

```
    public function get_items() {
        if (!$this->items) {
            $sql = "select * from Item "
                    . "where id in"
                    . " ("
                    . " select item from eventitem"
                    . " where event={$this->id}"
                    . ") order by type";
            $this->items = Item::find_by_sql($sql);
        }
        return $this->items;
    }

    public function find_remaining_items() {
        $sql = "select * from Item "
                . "where id not in"
                . " ("
                . " select item from eventitem"
                . " where event={$this->id}"
                . ")";
        $items = Item::find_by_sql($sql);
        return $items;
    }

    public function find_remaining_items_of_type($type) {
        $sql = "select * from Item "
                . "where type='{$type}' and id not in"
                . " ("
                . " select item from eventitem"
                . " where event={$this->id}"
                . ")";
        $items = Item::find_by_sql($sql);
        return $items;
    }

    public function save() {
        $organiser = User::find_by_id($this->organiser);
        if ($organiser) {
            parent::save();
        } else {
            die('Organiser is not valid');
        }
    }

    public function create() {
        parent::create();
        $invitation = Invitation::make($this->id, $this-
>organiser, "You created the event!");
        $invitation->position = 'Admin';
        $invitation->status = 'Attending';
        $invitation->save();
    }
```

```
    public static function find_for_user($user_id) {
        if (!empty($user_id) && $user_id != 0) {
            $sql = "select * from ." . static::$table_name
. " where organiser = $user_id and datetime >=
CURRENT_DATE() order by datetime desc";
            return static::find_by_sql($sql);
        }
    }

    public function name() {
        return $this->name;
    }

    public function get_organiser() {
        return User::find_by_id($this->organiser);
    }

    public function anchor($image_size = "72px", $class =
"", $classImg = "img img-thumbnail", $title = "org") {

        $anchor = "<a
href=\"./view_event.php?event={$this->id}\"
class=\"$class\" >"
                . $this->avatar($image_size, $classImg,
$title)
                . "<em>"
                . $this->name()
                . "</em>"
                . "</a>";

        return $anchor;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Event Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Name
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description of Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event Organiser
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Date Time
```

```
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class = \"col-sm-12 col-md-2\">" .
$this->name . "</td>"
                . "<td class = \"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                . "<td class = \"col-sm-12 col-md-2\">" .
$this->get_organiser()->avatar() . "</td>"
                . "<td class = \"col-sm-12 col-md-2\">" .
$this->datetime . "</td>"
                . "</tr>";
    }

}

?>
```

EventItem.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class EventItem extends DatabaseObject {

    protected static $table_name = "eventitem";
    protected static $db_fields = array('id', 'event',
'item', 'note');
    public $id;
    public $event;
    public $item;
    public $note;
    public $eventObj, $itemObj;

    public static function make($event, $item, $note) {
        $eventItem = new self;
        $eventItem->event = $event;
        $eventItem->item = $item;
        $eventItem->note = $note;
        return $eventItem;
    }

    public function name() {
        return "Nothing to show here";
```

```
    }

    public function init_members() {
        $this->eventObj = Event::find_by_id($this->event);
        $this->itemObj = Item::find_by_id($this->item);
    }

    public function get_event() {
        if (!$this->eventObj) {
            $this->init_members();
        }
        return $this->eventObj;
    }

    public function get_item() {
        if (!$this->itemObj) {
            $this->init_members();
        }
        return $this->itemObj;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Item
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Note
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_event()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_item()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->note . "</td>"
                . "</tr>";
    }

}
```

```
?>
```

EventReview.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class EventReview extends DatabaseObject {

    protected static $table_name = "eventreview";
    protected static $db_fields = array('id', 'event',
'user', 'title', 'description', 'posted_on', 'rating',
'img');
    public $id;
    public $event;
    public $user;
    public $title;
    public $description;
    public $posted_on;
    public $rating;
    public $eventObj, $userObj;
    public $img;

    public static function make($event, $user, $title,
$description, $posted_on, $rating) {
        $event_review = new self;
        $event_review->event = $event;
        $event_review->user = $user;
        $event_review->title = $title;
        $event_review->description = $description;
        $event_review->posted_on = $posted_on;
        $event_review->rating = $rating;

        return $event_review;
    }

    public function name() {
        return $this->title;
    }

    public static function find_numerically($event){
        global  $database;
        $sql = "select rating, count(rating) from
eventreview where event = {$event} GROUP by rating order
by rating desc";
        $ratings = array();
        $total = 0;
        $result = $database->query($sql);
        while($row = $database->fetch_array($result)){
            $ratings[$row[0]] = $row[1];
```

```
            $total = $total + $row[1];
        }
        $ratings["total"] = $total;
        return $ratings;
    }

    public static function find_all_for_event($event) {
        $sql = "select * from "
                . static::$table_name
                . " where event=$event";
        return self::find_by_sql($sql);
    }

    public static function
find_for_event_by_user($event_id, $user_id) {
        $sql = "select * from "
                . static::$table_name
                . " where event=$event_id"
                . " and user=$user_id";
        $review = self::find_by_sql($sql);
        return array_shift($review);
    }

    public function init_members() {
        $this->eventObj = Event::find_by_id($this->event);
        $this->userObj = User::find_by_id($this->user);
    }

    public function get_user() {
        if (!$this->userObj) {
            $this->init_members();
        }

        return $this->userObj;
    }

    public function get_event() {
        if (!$this->eventObj) {
            $this->init_members();
        }
        return $this->eventObj;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
```

```
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        User
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Posted On
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Rating
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->eventObj->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\"> " .
$this->userObj->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->posted_on . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->rating . "</td>
                </tr>";
    }

}

?>
```

EventVenue.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class EventVenue extends DatabaseObject {
```

```
    protected static $table_name = "eventvenue";
    protected static $db_fields = array('id', 'event',
'venue', 'bookedBy', 'bookedOn');
    public $id;
    public $event;
    public $venue;
    public $bookedBy;
    public $bookedOn;
    public $eventObj, $userObj, $venueObj;

    public static function make($event, $venue, $bookedBy)
{
        $eventVenue = new self;
        $eventVenue->event = $event;
        $eventVenue->venue = $venue;
        $eventVenue->bookedBy = $bookedBy;
        return $eventVenue;
    }

    public function name() {
        return "Nothing to show here";
    }

    public function init_members() {
        $this->userObj = User::find_by_id($this-
>bookedBy);
        $this->eventObj = Event::find_by_id($this->event);
        $this->venueObj = Venue::find_by_id($this->venue);
    }

    public function get_user() {
        if (!$this->userObj) {
            $this->init_members();
        }
        return $this->userObj;
    }

    public function get_event() {
        if (!$this->eventObj) {
            $this->init_members();
        }
        return $this->eventObj;
    }

    public function get_venue() {
        if (!$this->venueObj) {
            $this->init_members();
        }
        return $this->venueObj;
    }

    public function renderTableHeader() {
        return '
```

```
                <thead>
                    <tr class="row">
                        <th class="col-sm-12 col-md-2 ">
                            Booking Id
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Event
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Venue
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Booked By
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Booked On
                        </th>
                    </tr>
                </thead>';
        }

        public function renderTableRow($edit = TRUE) {
            return "<tr class=\"row\">"
                    . $this->table_edit($edit)
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_event()->intro() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_venue()->intro() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_user()->intro() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->bookedOn . "</td>"
                    . "</tr>";
        }

    }

    ?>
```

Gallery.php

```
    <?php

    // If it is going to need the database, then it is
    probably smart to require it before we start.
    require_once(LIB_PATH . DS . "database.php");

    class Gallery extends DatabaseObject {

        // this class is used for the database table named
    users.
        protected static $table_name = "gallery";
        protected static $db_fields = array('id', 'event',
    'note', 'img');
```

```
    public $id;
    public $event;
    public $note;
    public $img;
    public $eventObj;
    private $comments;

    public static function make($event, $note, $img) {
        $gallery = new Gallery();
        $gallery->event = $event;
        $gallery->note = $note;
        $gallery->img = $img;
    }

    public function validate_data() {
        $attributes = array('event', 'note', 'img');
        return $this->validate_attributes($attributes);
    }

    public function init_members() {
        if (!$this->eventObj) {
            $this->eventObj = Event::find_by_id($this-
>event);
        }
    }

    public function get_comments() {
        if (!$this->comments) {
            $this->comments =
ImageComment::find_for_image($this);
        }
        return $this->comments;
    }

    public static function find_all_for_event($event) {
        $sql = "select * from "
                . static::$table_name
                . " where event=$event";
        return self::find_by_sql($sql);
    }

    public function name() {
        return $this->note;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Image ID
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
```

```
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Note
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Event
                        </th>
                    </tr>
                </thead>';
        }

        public function renderTableRow($edit = TRUE) {
            $this->init_members();
            return "<tr class=\"row\">"
                    . $this->table_edit($edit)
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->avatar() . "</td>"
                    . "<td class = \"col-sm-12 col-md-2\">" .
    $this->note . "</td>"
                    . "<td class = \"col-sm-12 col-md-2\">" .
    $this->eventObj->avatar() . "</td>"
                    . "</tr>";
        }

    }

    ?>
```

Guest.php

```
    <?php

    // If it is going to need the database, then it is
    probably smart to require it before we start.
    require_once(LIB_PATH . DS . "database.php");

    class Guest extends DatabaseObject {

        protected static $table_name = "guest";
        protected static $db_fields = array('id', 'user',
    'event', 'position', 'status');
        public $id;
        public $user;
        public $event;
        public $position;
        public $status;

        public static function make($user, $event, $position,
    $status) {
            $guest = new self;
            $guest->user = $user;
            $guest->event = $event;
            $guest->position = $position;
            $guest->status = $status;
```

```
    return $guest;
}

public function get_user() {
    return User::find_by_id($this->user);
}

public function get_event() {
    return Event::find_by_id($this->event);
}

public static function find_all_for_event($event) {
    $sql = "select * from " . static::$table_name
            . " where event=$event";
    return static::find_by_sql($sql);
}

public static function is_guest($user_id, $event_id) {
    $sql = "select * from " . static::$table_name
            . " where user = '{$user_id}'"
            . " and event = '{$event_id}'";
    return self::find_by_sql($sql);
}

public function css_class() {
    switch ($this->status) {
        case 'Attending':
            return 'success';
        case 'Not Attending':
            return 'danger';
        case 'May Be':
            return 'warning';
    }
}

public function renderTableHeader() {
    return '
        <thead>
            <tr class="row">
                <th class="col-sm-12 col-md-2 ">
                    Id
                </th>
                <th class="col-sm-12 col-md-2 ">
                    Guest
                </th>
                <th class="col-sm-12 col-md-2 ">
                    Event
                </th>
                <th class="col-sm-12 col-md-2 ">
                    Position
                </th>
                <th class="col-sm-12 col-md-2 ">
                    Invitation Status
                </th>
```

```
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_user()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\"> " .
$this->get_event()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->position . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->status . "</td>"
                . "</tr>";
    }

}

?>
```

ImageComment.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class ImageComment extends DatabaseObject {

    protected static $table_name = "imagecomment";
    protected static $db_fields = array('id', 'image',
'user', 'comment', 'datetime');
    public $id;
    public $image;
    public $user;
    public $comment;
    public $datetime;
    public $imageObj, $userObj;

    public function init_members() {
        if (!$this->imageObj) {
            $this->imageObj = Item::find_by_id($this-
>image);
        }

        if (!$this->userObj) {
            $this->userObj = User::find_by_id($this-
>user);
        }
    }
```

```
    public static function find_for_image($image) {
        global $database;
        $sql = "select * from " . static::$table_name
                . " where"
                . " image = {$image->id}";
        $result_set = $database->query($sql);
        $object_array = array();
        while ($row = $database->fetch_array($result_set))
{
            $comment = static::instantiate($row);
            $comment->imageObj = $image;
            $comment->init_members();
            $object_array[] = $comment;
        }
        return $object_array;
    }

    public static function make($image, $user, $comment) {
        $image_comment = new self;
        $image_comment->image = $image;
        $image_comment->user = $user;
        $image_comment->comment = $comment;
        return $image_comment;
    }

    public function create() {
        global $database;
        $sql = "INSERT INTO " . static::$table_name . "
(";
        $sql .= "image,user,comment";
        $sql .= ") VALUES (";
        $sql .= "{$this->image},{$this->user},'{$this-
>comment}'";
        $sql .= ")";
        if ($database->query($sql)) {
            $this->id = $database->insert_id();
            return true;
        } else {
            return false;
        }
    }

    public function name() {
        return $this->comment;
    }

    public function title() {
        $this->init_members();
        return $this->userObj->intro();
    }

    public function renderTableHeader() {
        return '
            <thead>
```

```
                    <tr class="row">
                        <th class="col-sm-12 col-md-2 ">
                            Id
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Image
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            User
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Comment
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Date Time
                        </th>
                    </tr>
                </thead>';
        }

        public function get_user() {
            $this->init_members();
            return $this->userObj;
        }

        public function get_image() {
            $this->init_members();
            return $this->imageObj;
        }

        public function renderTableRow($edit = TRUE) {
            return "<tr class=\"row\">"
                    . $this->table_edit($edit)
                    . "<td class=\"col-sm-12 col-md-2\"> " .
    $this->get_image()->intro() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\"> " .
    $this->get_user()->intro() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->comment . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->datetime() . "</td>"
                    . "</tr>";
        }

    }

    ?>
```

Invitation.php

```
    <?php

    // If it is going to need the database, then it is
    probably smart to require it before we start.
```

```
require_once(LIB_PATH . DS . "database.php");

class Invitation extends DatabaseObject {

    protected static $table_name = "Invitation";
    protected static $db_fields = array('id', 'event',
'user', 'message', 'date');
    public $id;
    public $event;
    public $user;
    public $message;
    public $date;
    public $position, $status;
    public $eventObj, $userObj, $guestObj;

    public static function make($event, $user, $message) {
        $invitation = new Invitation();
        $invitation->event = $event;
        $invitation->user = $user;
        $invitation->message = $message;
        $invitation->date = date("Y-m-d") . 'T' .
date("h:i:s");
        return $invitation;
    }

    public static function find_all_for_user($user) {
        $sql = "select * from "
                . static::$table_name
                . " where user=$user order by date desc,
id desc";
        return Invitation::find_by_sql($sql);
    }

    public static function find_upcoming($user) {
        $sql = "select * from "
                . static::$table_name
                . " where user=$user "
                . " and event in (select id from event
where datetime >= CURRENT_DATE())"
                . " order by date desc, id desc";
        return Invitation::find_by_sql($sql);
    }
    public static function find_past($user) {
        $sql = "select * from "
                . static::$table_name
                . " where user=$user "
                . " and event in (select id from event
where datetime < CURRENT_DATE())"
                . " order by date desc, id desc";
        return Invitation::find_by_sql($sql);
    }

    public function create() {
        parent::create();
```

```
        $this->init_members();
        $this->guestObj = Guest::make($this->user, $this-
>event, $this->position, $this->status);
        $this->guestObj->save();
    }

    public function update() {
        parent::update();
        $this->init_members();
        $this->guestObj->position = $this->position;
        $this->guestObj->status = $this->status;
        $this->guestObj->save();
    }

    public function init_members() {
        if ($this->event && $this->user) {
            $this->eventObj = Event::find_by_id($this-
>event);
            $this->userObj = User::find_by_id($this-
>user);
            $obj = Guest::find_by_sql(
                        "SELECT * FROM " . "guest"
                        . " where user=" . $this->user
                        . " and event=" . $this->event
                        . " order by id desc"
            );
            $this->guestObj = array_shift($obj);
            if (empty($this->status)) {
                $this->status = $this->guestObj->status;
            }
            if (empty($this->position)) {
                $this->position = $this->guestObj-
>position;
            }
        } else {
            $this->eventObj = new Event();
            $this->userObj = new User();
            $this->guestObj = new Guest();
        }
    }

    public function name() {
        $this->init_members();
        return "<div>"
                . ""
                . $this->eventObj->avatar()
                . $this->message
                . "</div>"
        ;
    }

    public function status() {
        switch ($this->status) {
            case 'Attending':
```

```
                $class = "success";
                $msg = "Looking forward to the event with
you.";
                break;
            case 'Not Attending':
                $class = "danger";
                $msg = "Would have been nice of you to
come.";
                break;

            case 'May Be':
                $class = "warning";
                $msg = "Awaiting response.";
                break;
        }
        return "<p class=\"text-$class\">$msg</p>";
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        User
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Message
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Position
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Status
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->eventObj->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->userObj->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->message . "</td>"
```

```
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->guestObj->position . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->guestObj->status . "</td>"
                . "</tr>";
    }

}

?>
```

Item.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class Item extends DatabaseObject {

    protected static $table_name = "Item";
    protected static $db_fields = array('id', 'title',
'description', 'type', 'img');
    public $id;
    public $title;
    public $description;
    public $type;
    public $img;
    public $note;

    public static function make($title, $description,
$type) {
        $item = new self;
        $item->title = $title;
        $item->description = $description;
        $item->type = $type;
        return $item;
    }

    public function name() {
        return $this->title;
    }

    public function title() {
        return $this->description;
    }

    public function note($event) {
        if (!$this->note) {
            $sql = "select * from eventitem where
event=$event and item=$this->id";
            $eventItem = EventItem::find_by_sql($sql);
            $eventItem = array_shift($eventItem);
```

```
            $this->note = $eventItem->note;
        }
        return $this->note;
    }

    public static function get_available_types() {
        global $database;
        $sql = "select distinct type from item";
        $res = $database->query($sql);
        while (($row = mysqli_fetch_assoc($res))) {
            $objects[] = array_shift($row);
        }
        return $objects;
    }

    public static function find_ordered() {
        global $database;
        $sql = "select * from item order by type";
        $res = $database->query($sql);
        $objects = array();
        while ($row = mysqli_fetch_assoc($res)) {
            $type = strtolower($row['type']);
            $objects[$type][] = $row;
        }
        return $objects;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Type
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
```

```
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->type . "</td>"
                . "</tr>";
    }

}

?>
```

ItemReview.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class ItemReview extends DatabaseObject {

    protected static $table_name = "itemreview";
    protected static $db_fields = array('id', 'event',
'item', 'user', 'title', 'description', 'posted_on',
'rating', 'img');
    public $id;
    public $event;
    public $item;
    public $user;
    public $title;
    public $description;
    public $posted_on;
    public $rating;
    public $img;
    public $itemObj, $userObj, $eventObj;

    public function init_members() {
        if (!$this->itemObj) {
            $this->itemObj = Item::find_by_id($this-
>item);
        }

        if (!$this->eventObj) {
            $this->eventObj = Event::find_by_id($this-
>event);
        }

        if (!$this->userObj) {
            $this->userObj = User::find_by_id($this-
>user);
```

```
        }
    }

    public static function make($event, $item, $user,
$title, $description, $posted_on, $rating) {
        $item_review = new self;
        $item_review->event = $event;
        $item_review->item = $item;
        $item_review->user = $user;
        $item_review->title = $title;
        $item_review->description = $description;
        $item_review->posted_on = $posted_on;
        $item_review->rating = $rating;

        return $item_review;
    }

    public function name() {
        return $this->title;
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Item
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        User
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Posted On
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Rating
                    </th>
                </tr>
            </thead>';
```

```
    }

    public function get_user() {
        $this->init_members();
        return $this->userObj;
    }

    public function get_event() {
        $this->init_members();
        return $this->eventObj;
    }

    public function get_item() {
        $this->init_members();
        return $this->itemObj;
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class = \"col-sm-12 col-md-2\">" .
$this->get_event()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\"> " .
$this->get_item()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\"> " .
$this->get_user()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->posted_on . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->rating . "</td>"
                . "</tr>";
    }

}

?>
```

Schedule.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class Schedule extends DatabaseObject {

    protected static $table_name = "schedule";
```

```
    protected static $db_fields = array('id', 'event',
'title', 'description', 'datetime');
    public $id;
    public $event;
    public $title;
    public $description;
    public $datetime;

    public static function make($event, $title,
$description, $datetime) {
        $schedule = new self;
        $schedule->event = $event;
        $schedule->title = $title;
        $schedule->description = $description;
        $schedule->datetime = $datetime;

        return $schedule;
    }
    public static function find_all_for_event($event) {
        $sql = "select * from "
                . static::$table_name
                . " where event=$event";
        return Schedule::find_by_sql($sql);
    }

    public function name(){
        return $this->title;
    }

    public function get_event() {
        return Event::find_by_id($this->event);
    }


    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Event
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Date Time
                    </th>
                </tr>
```

```
                </thead>';
        }

        public function renderTableRow($edit = TRUE) {
            return "<tr class=\"row\">"
                    . $this->table_edit($edit)
                    . "<td>" . $this->get_event()->intro() .
"</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
$this->datetime . "</td>"
                    . "</tr>";
        }

    }

    ?>
```

Session.php

```
    <?php

    /*
      A class to work with SESSIONs
      In our case, primarily to manage logging users in and
    out

      Keep in mind when working with sessions that it is
    generally inadvisable to store DB-related objects in
    sessions.
     */

    class Session {

        private $logged_in = false;
        public $user_id;
        public $user_type;
        public $message;
        public $request_uri;

        function __construct() {
            session_start();
            $this->check_message();
            $this->check_login();
            $this->check_request_uri();

            if ($this->logged_in) {
                // TODO actions to take right away if user is
    logged in
            } else {
```

```
            // TODO actions to take right away if user is
not logged in
        }
    }

    public function is_logged_in() {
        return $this->logged_in;
    }

    public function login($user) {
        // database should find user based on
username/password
        if ($user) {
            $this->user_id = $_SESSION['user_id'] = $user-
>id;
            $this->user_type = $_SESSION['user_type'] =
$user->type;
            $this->logged_in = true;
        }
    }

    public function logout() {
        unset($_SESSION['user_id']);
        unset($this->user_id);
        $this->logged_in = false;
    }

    private function check_login() {
        if (isset($_SESSION['user_id'])) {
            $this->user_id = $_SESSION['user_id'];
            $this->logged_in = true;
        } else {
            unset($this->user_id);
            $this->logged_in = false;
        }
    }

    private function check_message() {
        // Is there a message stored in the session?
        if (isset($_SESSION['message'])) {
            // Add it as an attribute and erase the stored
version
            $this->message = $_SESSION['message'];
            unset($_SESSION['message']);
        } else {
            $this->message = "";
        }
    }

    private function check_request_uri() {
        if (isset($_SESSION['request_uri'])) {
            $this->request_uri = $_SESSION['request_uri'];
            unset($_SESSION['request_uri']);
        } else {
```

```
            $this->request_uri = "";
        }
    }

    public function message($msg = "") {
        if (!empty($msg)) {
            // then this is "set message"
            // make sure you understand why $this->message
= $msg would not work
            $_SESSION['message'] = $msg;
        } else {
            // then this is 'get message'
            return $this->message;
        }
    }

    public function request_uri($uri = "") {
        if (!empty($uri)) {
            // then this is "set message"
            // make sure you understand why $this->message
= $msg would not work
            $_SESSION['request_uri'] = $uri;
        } else {
            // then this is 'get message'
            return $this->request_uri;
        }
    }

    public function get_user_object() {
        if (isset($this->user_id)) {
            $user = User::find_by_id($this->user_id);
            return $user;
        } else {
            return new User();
        }
    }

    public function is_admin() {
        return $this->get_user_object()->is_admin();
    }

}

$session = new Session();
$message = $session->message();
$request_uri = $session->request_uri();
?>
```

Task.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
```

```
require_once(LIB_PATH . DS . "database.php");

class Task extends DatabaseObject {

    protected static $table_name = "task";
    protected static $db_fields = array('id', 'checklist',
'title', 'details', 'status', 'deadline', 'img',
'assigned_to');
    public $id;
    public $checklist;
    public $title;
    public $details;
    public $status;
    public $deadline;
    public $img;
    public $assigned_to;
    public $checklistObj, $userObj;

    public static function make($checklist, $title,
$details, $status, $deadline) {
        $task = new self;
        $task->checklist = $checklist;
        $task->title = $title;
        $task->details = $details;
        $task->status = $status;
        $task->deadline = $deadline;

        return $task;
    }

    public function name() {
        return $this->title;
    }

    public function deadline($format="h:i a, F d Y"){
        return static::format_datetime($format, $this-
>deadline);
    }

    public function init_members() {
        $this->checklistObj = CheckList::find_by_id($this-
>checklist);
        $this->checklistObj->init_members();
        $this->userObj = User::find_by_id($this-
>assigned_to);
    }

    public function get_user() {
        if (!$this->userObj) {
            $this->init_members();
        }
        return $this->userObj;
    }
```

```
    public function get_checklist() {
        if (!$this->checklistObj) {
            $this->init_memebers();
        }
        return $this->checklistObj;
    }

    public static function
find_all_for_checklist($checklist) {
        $sql = "select * from "
                . static::$table_name
                . " where checklist=$checklist";
        return Task::find_by_sql($sql);
    }

    public function get_class() {
        $status = strtolower($this->status);

        switch ($status) {
            case 'assigned':
                return 'warning';
            case 'completed':
                return 'success';
            case 'failed':
                return 'danger';
            case 'started':
            case 'working':
            default :
                return 'default';
        }
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Task Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Check List
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Title
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Details
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Assigned To
                    </th>
```

```
                    <th class="col-sm-12 col-md-2 ">
                        Status
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Deadline
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_checklist()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->title . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->details . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->get_user()->intro() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->status . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->deadline . "</td>"
                . "</tr>";
    }

}

?>
```

User.php

```
<?php

// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class User extends DatabaseObject {

    protected static $table_name = "user";
    protected static $db_fields = array('id',
'first_name', 'last_name', 'user_name', 'password',
'address', 'email', 'img', 'type');
    public $id;
    public $first_name;
    public $last_name;
    public $user_name;
    public $password;
    public $address;
```

```
    public $email;
    public $img;
    public $type;

    public static function make($first_name, $last_name,
$user_name, $password, $address, $email) {
        $user = new self;
        $user->first_name = $first_name;
        $user->last_name = $last_name;
        $user->user_name = $user_name;
        $user->password = $password;
        $user->address = $address;
        $user->email = $email;
        return $user;
    }

    public function is_admin() {
        $type = strtolower($this->type);
        return ($type == "admin");
    }

    public function name() {
        return "$this->first_name $this->last_name";
    }

    public static function find_uninvited($event) {
        $sql = "select * from user"
                . " where"
                . " id not in"
                . " (select user from guest where event =
$event)";
        return static::find_by_sql($sql);
    }

    public static function find_members_of_event($event) {
        $sql = "select * from user"
                . " where"
                . " id in"
                . " (select user from guest where event =
$event and position in('Admin','Member'))";
        return static::find_by_sql($sql);
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
                    <th class="col-sm-12 col-md-2 ">
                        Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
```

```
                            First Name
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Last Name
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            User Name
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Password
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Address
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Email
                        </th>
                        <th class="col-sm-12 col-md-2 ">
                            Type
                        </th>
                    </tr>
                </thead>';
        }

        public function renderTableRow($edit = TRUE) {
            return "<tr class=\"row\">"
                    . $this->table_edit($edit)
                    . "<td class=\"col-sm-12 col-md-2\"> " .
    $this->avatar() . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->first_name . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->last_name . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->user_name . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->password . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->address . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->email . "</td>"
                    . "<td class=\"col-sm-12 col-md-2\">" .
    $this->type . "</td>"
                    . "</tr>";
        }

    }

?>
```

Venue.php

```
    <?php
```

```
// If it is going to need the database, then it is
probably smart to require it before we start.
require_once(LIB_PATH . DS . "database.php");

class Venue extends DatabaseObject {

    protected static $table_name = "venue";
    protected static $db_fields = array('id', 'name',
'address', 'description', 'capacity', 'img');
    public $id;
    public $name;
    public $address;
    public $description;
    public $capacity;
    public $img;

    public static function make($name, $address,
$description, $capacity) {
        $venue = new self;
        $venue->name = $name;
        $venue->address = $address;
        $venue->description = $description;
        $venue->capacity = $capacity;
        return $venue;
    }

    public function name() {
        return $this->name;
    }

    public static function find_available($date){
        $sql = "SELECT * from venue"
                . " where id not in "
                . "(select venue from eventvenue where"
                . " event in "
                . "(select id from event where
event.datetime like '%$date%')"
                . ")";
        return Venue::find_by_sql($sql);
    }

    public static function find_for_event($event) {
        $sql = "select * from "
                . static::$table_name
                . " where id in (select venue from
eventvenue where event=$event)";
        $venue = Venue::find_by_sql($sql);
        return array_shift($venue);
    }

    public function renderTableHeader() {
        return '
            <thead>
                <tr class="row">
```

```
                    <th class="col-sm-12 col-md-2 ">
                        Venue Id
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Image
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Name
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Description
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Capacity
                    </th>
                    <th class="col-sm-12 col-md-2 ">
                        Address
                    </th>
                </tr>
            </thead>';
    }

    public function renderTableRow($edit = TRUE) {
        return "<tr class=\"row\">"
                . $this->table_edit($edit)
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->avatar() . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->name . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->description . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->capacity . "</td>"
                . "<td class=\"col-sm-12 col-md-2\">" .
$this->address . "</td>"
                . "</tr>";
    }

}

?>
```

Initialize.php

```
<?php

// Web site constants
defined('DS') ? null : define('DS', '\\');
//defined('SITE_ROOT') ? null : define('SITE_ROOT',
$_SERVER['DOCUMENT_ROOT']);
defined('SITE_ROOT') ? null : define('SITE_ROOT', "C:" .
DS . "wamp64" . DS . "www" . DS . "events");
defined('LIB_PATH') ? null : define('LIB_PATH', SITE_ROOT
. DS . 'includes');
```

```
defined('UPLOAD_PATH') ? null : define('UPLOAD_PATH',
SITE_ROOT . DS . 'uploads');

if (!is_dir(UPLOAD_PATH)) {
    mkdir(UPLOAD_PATH, 0777, true);
}
// load configuration file
require_once(LIB_PATH . DS . "config.php");

// load basic functions
require_once(LIB_PATH . DS . "functions.php");

// load core objects
require_once(LIB_PATH . DS . "session.php");
require_once(LIB_PATH . DS . "database.php");
require_once(LIB_PATH . DS . "database_object.php");

// load database-related classes
require_once(LIB_PATH . DS . "checklist.php");
require_once(LIB_PATH . DS . "event.php");
require_once(LIB_PATH . DS . "eventReview.php");
require_once(LIB_PATH . DS . "guest.php");
require_once(LIB_PATH . DS . "item.php");
require_once(LIB_PATH . DS . "itemReview.php");
require_once(LIB_PATH . DS . "schedule.php");
require_once(LIB_PATH . DS . "task.php");
require_once(LIB_PATH . DS . "user.php");
require_once(LIB_PATH . DS . "venue.php");
?>
```

Business Logic

Attend.php

```
<?php

header("Content-type: application/json"); // Adding a
content type helps as well

include '../includes/initialize.php';
$current_user = $session->get_user_object();
$invitation =
Invitation::find_by_id($_POST['invitation_id']);
$invitation->init_members();
if ($_POST['status'] == '1') :
    $invitation->status = 'Attending';

else:
    $invitation->status = 'Not Attending';
endif;

$invitation->save();
$invitation->msg = $invitation->status();
echo json_encode($invitation);
?>
```

Comment.php

```
<?php

require_once './../includes/initialize.php';
$image = (isset($_POST['image'])) ? $_POST['image'] :
false;
$comment = (isset($_POST['comment'])) ? $_POST['comment']
: false;
$user = $session->get_user_object();
$user_id = ($user->id) ? $user->id : false;


if ($image && $comment && $user_id) {
    $image_comment = ImageComment::make($image, $user_id,
$comment);
    $image_comment->save();
    if ($image_comment->id) {
        $session->message("Comment Posted");
    } else {
        $session->message("Could not upload : " .
$comment);
    }
} else {
    $session->message("Oops there was a problem with your
request to post: " . $comment);
}
$redirect_url = (isset($_POST['redirect_url'])) ?
$_POST['redirect_url'] : "../index.php";
redirect_to($redirect_url);
```

Create_Event.php

```
<?php

require_once './../includes/initialize.php';

$event = new Event();
$attrs = $event->attributes();

foreach ($_POST as $attribute => $value) {
    if ($attribute == "table_name" || $attribute == "id")
{
        continue;
    }
    if (array_key_exists($attribute, $attrs) &&
!empty($value)) {
        $event->$attribute = $value;
    } else if (property_exists($event, $attribute)) {
        $event->$attribute = $value;
    }
}
```

```
if ($event->validate_attributes($event-
>insertion_attributes())) {

    if (!empty($_FILES['img']['name'])) {
        $event->img = $event->upload_img($_FILES['img']);
    }
    $event->save();
    if ($event->id) {
        $event_type = isset($_POST['event_type']) ?
$_POST['event_type'] : 0;

        $agenda =
isset($_POST[Created_Checklists::AGENDA]) ?
get_checkbox_value($_POST[Created_Checklists::AGENDA]) :
false;
        $artist =
isset($_POST[Created_Checklists::ARTIST]) ?
get_checkbox_value($_POST[Created_Checklists::ARTIST]) :
false;
        $decorations =
isset($_POST[Created_Checklists::DECORATIONS]) ?
get_checkbox_value($_POST[Created_Checklists::DECORATIONS]
) : false;
        $guest = isset($_POST[Created_Checklists::GUEST])
? get_checkbox_value($_POST[Created_Checklists::GUEST]) :
false;
        $legal = isset($_POST[Created_Checklists::LEGAL])
? get_checkbox_value($_POST[Created_Checklists::LEGAL]) :
false;
        $lights =
isset($_POST[Created_Checklists::LIGHTS]) ?
get_checkbox_value($_POST[Created_Checklists::LIGHTS]) :
false;
        $sound = isset($_POST[Created_Checklists::SOUND])
? get_checkbox_value($_POST[Created_Checklists::SOUND]) :
false;
        $visuals =
isset($_POST[Created_Checklists::VISUAL]) ?
get_checkbox_value($_POST[Created_Checklists::VISUAL]) :
false;

        $checkLists =
Created_Checklists::checklist_organisational($event,
$legal, $agenda, $sound, $lights, $guest, $visuals,
$decorations, $artist);
        while ($m_c_list = current($checkLists)) {
            $m_c_list->save();
            next($checkLists);
        }


        $items = array();
        $selected_items = (isset($_POST['item'])) ?
$_POST['item'] : false;
```

```
        if ($selected_items) {
            while ($item_select =
current($selected_items)) {
                $my_item = EventItem::make($event->id,
$item_select, "Pre selected in package.");
                $my_item->save();
                $items[] = $my_item;
                next($selected_items);
            }
        }
    }
    $redirect_url = "../view_event.php?event={$event-
>id}";

//    echo $redirect_url;
//    echo '<pre>';
//    print_r($_POST);
//    echo '</pre>';

//    echo '<pre>';
//    print_r($checkLists);
//    print_r($items);
//    echo '</pre>';
    redirect_to($redirect_url);
} else {
    echo 'Errors';
}
```

Invite_Guests.php

```
<?php
require_once('../includes/initialize.php');
global $session;
$table = $_POST["table_name"];
$users = $_POST['users'];
$invitations = array();
while ($guest = current($users)) {
    $object = Invitation::make($_POST['event'], $guest,
$_POST['message']);
    $object->position = $_POST['position'];
    $object->status = $_POST['status'];
    if ($object->validate_attributes($object-
>insertion_attributes())) {
        $object->save();
    }
    next($users);
}
$redirect_url = (isset($_POST['redirect_url'])) ?
$_POST['redirect_url'] :
"../listTables.php?table={$table}";
redirect_to($redirect_url);
?>
```

Item_Options.php

```
<?php

include '../includes/initialize.php';
if (isset($_POST['event'])) {
    $type = (isset($_POST['type'])) ? $_POST['type'] : "-
";
    $current_id = $_POST['event'];
    if ($current_id > 0) {
        $current_event = Event::find_by_id($current_id);
    } else {
        $current_event = new Event();
        $current_event->id = 0;
    }

    if ($type != "-") {
        $options = $current_event-
>find_remaining_items_of_type($type);
    } else {
        $options = $current_event->find_remaining_items();
    }

    include '../layouts/data/options_list.php';
}
?>
```

Modify_Task.php

```
<?php

header("Content-type: application/json"); // Adding a
content type helps as well

require_once('../includes/initialize.php');
global $session;
$task_id = $_POST["task_id"];
$status = $_POST['status'];
$task = Task::find_by_id($task_id);
$task->status = $status;
$task->save();
echo json_encode($task);
```

HTML Layouts

Header.php

```
<!Doctype html>
<?php
require_once('includes/initialize.php');
global $session;
$current_user = $session->get_user_object();
if ($current_user->id > 0) {
    $invitations =
Invitation::find_all_for_user($current_user->id);
} else {
```

```
    $invitations = 0;
}
?>
<html lang="en">
    <head>
        <!--HTML specific data-->
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible"
content="IE=edge">

        <meta name="viewport" content="width=device-width,
initial-scale=1">

        <link rel="stylesheet"
href="custom/dist/css/bootstrap.min.css">
        <link rel="stylesheet"
href="custom/dist/css/bootstrap-theme.min.css">
        <link rel="stylesheet"
href="custom/dist/css/select2.css">
        <link rel="stylesheet"
href="custom/dist/css/lakshay.css">


        <!-- JavaScript libraries -->

        <script src="custom/dist/js/jquery-
3.1.1.min.js"></script>
        <script
src="custom/dist/js/bootstrap.min.js"></script>
        <script src="custom/dist/js/select2.js"></script>
        <script src="custom/dist/js/lakshay.js"></script>
        <script
src="custom/dist/js/jquery.validate.min.js"></script>
        <script src="custom/dist/js/jsrender.js"></script>


        <!--Site Specific Data-->
        <title><?php echo (!empty($page_title)) ?
$page_title : SITE_TITLE; ?></title>
        <meta name="theme-color" content="#101010">
        <link rel="icon" sizes="192x192"
href="images/grapes.png">

    </head>
    <body>
        <?php if (isset($custom_header) &&
$custom_header): ?>
        <?php else: ?>
            <?php include 'site_header.php'; ?>
        <?php endif; ?>

        <?php if ($session->message()): ?>
            <div class="alert alert-info alert-dismissible
fade in">
```

```
                <a href="#" class="close" data-
dismiss="alert" aria-label="close">&times;</a>
                <span class="text-center">
                    <strong>Message: </strong>
                    <?php echo $session->message(); ?>
                </span>
            </div>
        <?php endif; ?>
```

Navigation_Main.php

```
<nav class="navbar navbar-inverse navbar-static-top">
    <div class="navbar-header">
        <a class="navbar-toggle" data-toggle="collapse"
data-target="#site_nav">
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
        </a>
        <a class="navbar-brand" href="./">
            <?php echo SITE_TITLE; ?>
        </a>
    </div>
    <div id="site_nav" class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
            <?php
            include 'navigation_base.php';
            if (!$session->is_logged_in()):
                ?>
                <li class="navbar-right">
                    <a href="login.php"><span
class="glyphicon glyphicon-log-in"></span> Login</a>
                </li>
                <?php
            else:
                $user = $session->get_user_object();
                include 'navigation_user.php';
                if ($user->type == 'admin') {
                    include 'navigation_admin.php';
                }
                ?>
            <?php endif; ?>
        </ul>
    </div>
</nav>
```

Navigation_Base.php

```
<li>
    <a href="about.php#site"><span class="glyphicon
glyphicon-info-sign"></span> About</a>
</li>
```

Navigation_User.php

```
<li>
    <a href="./view_event.php"><span class="glyphicon
glyphicon-calendar"></span> Events</a>
</li>

<li class="navbar-right">
    <a href="./logout.php"><span class="glyphicon
glyphicon-log-out"></span> Logout <?php echo $user-
>first_name; ?></a>
</li>
<li>
    <!--<a onclick="showPopup(this.href);return(false);"
href="./invitations.php"><span class="glyphicon glyphicon-
bell"></span> Invitations</a>-->
    <a href="#" data-toggle="modal" data-
target="#myNotifications">
        <span class="glyphicon glyphicon-bell"></span>
Invitations <span id="notificationsCount"
class="badge"><?php echo sizeof($invitations); ?></span>
    </a>
</li>
```

Navigation_Admin.php

```
<li class="dropdown">
    <a class="dropdown-toggle" data-toggle="dropdown"
href="#">
        <span class="glyphicon glyphicon-wrench"></span>
Administration Tasks
        <!--<span class="caret"></span>-->
    </a>
    <ul class="dropdown-menu">
        <li><a href="list_tables.php?table=user"><span
class="glyphicon glyphicon-list-alt"></span> Manage
Tables</a></li>
    </ul>
</li>
```

Site_Branding.php

```
<?php
if (!isset($about_width)) {
    $about_width = 4;
}
?>

<div class="col-md-<?php echo $about_width;?>">
    <h2>About this site.</h2>
    <hr>
    <p>
        This site primarily deals with managing events and
all the tasks related to them.
    </p>
```

```
    <a class="btn btn-default" href="about.php">More
Info</a>
</div>
<!-- /.col-md-4 -->
<div class="col-md-<?php echo $about_width;?>">
    <h2>About the Developer
<em><small>(Me)</small></em></h2>
    <hr>
    <p>
        I <em><?php echo DEVELOPER_NAME; ?></em> am
persuing my Masters of Computer Applications from IGNOU
and in the final semester of it.
    </p>
    <a class="btn btn-default"
href="about.php#developer">More Info</a>
</div>
<!-- /.col-md-4 -->
<div class="col-md-<?php echo $about_width;?>">
    <h2>Some techy stuff</h2>
    <hr>
    <p>
        The site has been developed with the help of
Apache MySQL and PHP.Also a little bit of Bootstrap3 and
JQuery is used to make the site look beautiful.
    </p>
    <a class="btn btn-default" href="about.php#tech">More
Info</a>
</div>
<!-- /.col-md-4 -->
```

Site_Logo.php

```
<?php
include './images/lakshay.svg';

if (!isset($animation_duration)) {
    $animation_duration = 1750;
}
?>
<script src="custom/dist/js/jquery.drawsvg.js"></script>
<script type="text/javascript">
    $svg = $('.site_logo > svg').drawsvg({
        duration: <?php echo $animation_duration; ?>,
        callback: function () {
            $('.site_logo').addClass('active');
        }
    });

    function animateLogo() {
        $svg.drawsvg('animate');
    }

    animateLogo();
```

```
</script>
```

Site_Slider.php

```
<div id="myCarousel" class="carousel slide center-block"
data-ride="carousel">
    <!-- Indicators -->
    <ol class="carousel-indicators">
        <li data-target="#myCarousel" data-slide-to="0"
class="active"></li>
        <li data-target="#myCarousel" data-slide-
to="1"></li>
        <li data-target="#myCarousel" data-slide-
to="2"></li>
        <li data-target="#myCarousel" data-slide-
to="3"></li>
        <li data-target="#myCarousel" data-slide-
to="4"></li>
    </ol>

    <!-- Wrapper for slides -->
    <div class="carousel-inner" role="listbox">
        <div class="item active">
            <img class="img img-responsive"
src="images/5.jpg">
            <div class="carousel-caption">
                <!--                <h1>Where we go</h1>
                                <p>
                                    Journey is more
important than destination.
                                </p>-->
            </div>

        </div>

        <div class="item">
            <img class="img img-responsive"
src="images/5.jpg">
        </div>

        <div class="item">
            <img class="img img-responsive"
src="images/5.jpg">
        </div>

        <div class="item">
            <img class="img img-responsive"
src="images/5.jpg">
        </div>

        <div class="item">
            <img class="img img-responsive"
src="images/5.jpg">
        </div>
```

```
    </div>

    <!-- Left and right controls -->
    <a class="left carousel-control" href="#myCarousel"
role="button" data-slide="prev">
        <span class="glyphicon glyphicon-chevron-left"
aria-hidden="true"></span>
        <span class="sr-only">Previous</span>
    </a>
    <a class="right carousel-control" href="#myCarousel"
role="button" data-slide="next">
        <span class="glyphicon glyphicon-chevron-right"
aria-hidden="true"></span>
        <span class="sr-only">Next</span>
    </a>
</div>
```

Table_Render.php

```
<table class="table table-striped ">

    <?php

    $edit_records = (isset($edit_records)) ? $edit_records
: TRUE;
    $renderRow = false;
    if (method_exists(current($table_records),
"renderTableHeader")) {
        $renderRow = true;
        echo current($table_records)->renderTableHeader();
    } else {
        ?>
        <thead>
            <tr class="row">
                <?php
                foreach ($table_records[0]-
>get_db_fields() as $key):
                    ?>
                    <th class="col-sm-12 col-md-2 ">
                        <?php
                        echo ucwords($key);
                    endforeach;
                    ?>
                </th>
            </tr>
        </thead>
        <tbody>
            <?php
        }
        while ($row = current($table_records)) :
            ?>
            <?php
            if (method_exists($row, "renderTableRow") &&
$renderRow) {
```

```
                    echo $row->renderTableRow($edit_records);
                } else {
                    // Render generic version
                    ?>
                    <tr class="row">
                        <?php foreach ($row as $value): ?>
                            <?php ?>
                            <td class="col-sm-12 col-md-2">
                                <?php echo $value; ?>
                            </td>

                        <?php endforeach; ?>
                    </tr>
                    <?php
                }
                next($table_records);
            endwhile;
            ?>
        </tbody>
    </table>
```

Footer.php

```
    <?php if (isset($custom_footer) && $custom_footer): ?>
    <?php else: ?>
        <hr>
        <footer class="container-fluid" id="page_footer">
            <address class="col-md-3">
                <h3><?= DEVELOPER_NAME; ?></h3>
                <p>
                    House No 8
                    Mohalla No 14
                </p>
                Jalandhar Cantt
                144005
                <br/>
                <br/>

                <a class="btn btn-primary" href="mailto:<?=
    DEVELOPER_MAIL; ?>?subject=Contact for Support&body=Sent
    from <?= $_SERVER['REMOTE_ADDR']; ?>">
                    <span class="glyphicon glyphicon-
    envelope"></span>
                    <strong><?= DEVELOPER_NAME; ?></strong>
                </a>
            </address>

            <div class="col-md-6 col-md-offset-3">
                <div class="row">
                    <h3>Follow Updates on Social Media</h3>
                </div>
                <div class="row">
                    <a
    href="https://twitter.com/lakshay__verma" class="twitter-
```

```
follow-button" data-show-count="false" data-
size="large">Follow @lakshay__verma</a><script async
src="//platform.twitter.com/widgets.js" charset="utf-
8"></script>
            </div>
            <div class="row">
                <div id="fb-root"></div>
                <script>(function (d, s, id) {
                        var js, fjs =
d.getElementsByTagName(s)[0];
                        if (d.getElementById(id))
                            return;
                        js = d.createElement(s);
                        js.id = id;
                        js.src =
"//connect.facebook.net/en_US/sdk.js#xfbml=1&version=v2.8"
;
                        fjs.parentNode.insertBefore(js,
fjs);
                    }(document, 'script', 'facebook-
jssdk'));</script>
                <div class="fb-like" data-
href="https://facebook.com/lk.lakshay" data-
layout="standard" data-action="like" data-
colorscheme="dark" data-size="large" data-show-
faces="true" data-share="true"></div>
            </div>
            <br>


            <div class="row">
                <!-- Place this tag in your head or just
before your close body tag. -->
                <script
src="https://apis.google.com/js/platform.js" async
defer></script>

                <!-- Place this tag where you want the +1
button to render. -->
                <div class="g-plusone" data-size="tall"
data-annotation="none" data-
href="https://plus.google.com/u/0/+LakshayVermaS"></div>
            </div>

            <div class="row">
                <a class="wordpress-follow-button btn btn-
default" href="https://vermalakshay.wordpress.com" data-
blog="https://vermalakshay.wordpress.com" data-lang="en"
data-show-follower-count="true">Follow Lakshay Talks on
WordPress.com</a>
                <script type="text/javascript">(function
(d) {
```

```
                        var f =
d.getElementsByTagName('SCRIPT')[0], p =
d.createElement('SCRIPT');
                        p.type = 'text/javascript';
                        p.async = true;
                        p.src =
'//widgets.wp.com/platform.js';
                        f.parentNode.insertBefore(p, f);
                    }(document));</script>
            </div>
        </div>
    </footer>

<?php endif; ?>

<script>
    $('select').select2();
</script>


<!-- Notifications -->
<div id="myNotifications" class="modal fade"
role="dialog">
    <div class="modal-dialog">
        <!-- Modal content-->
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-
dismiss="modal">&times;</button>
                <h4 class="modal-title">Pending
Invitations</h4>
            </div>
            <div class="modal-body">
                <ul class="list-group">
                    <?php
                    $notifications_count = 0;
                    while ($invitation =
current($invitations)) :
                        $invitation->init_members();
                        if ($invitation->status == 'May
Be'):
                            ?>
                            <li class="list-group-item"
id="invite-<?php echo $invitation->id; ?>">
                                <?php echo $invitation-
>eventObj->avatar("48px", "img img-thumbnail zoom-img");
?>
                                <strong><?php echo
$invitation->eventObj->name(); ?></strong>
                                <p><?php echo $invitation-
>message; ?></p>
                                <div class="response">
```

```
                                        <a class="btn btn-sm
btn-success" onclick="attend_event(<?php echo $invitation-
>id; ?>, 1)">
                                            Attending
                                        </a>
                                        <a class="btn btn-sm
btn-default" onclick="attend_event(<?php echo $invitation-
>id; ?>, 0)">
                                            Not Attending
                                        </a>
                                    </div>
                                </li>
                                <?php
                                $notifications_count++;
                            endif;
                            next($invitations);
                        endwhile;
                        ?>

                    </ul>
                </div>
            </div>
            <script>
                notifications(<?php echo $notifications_count;
?>);
            </script>
        </div>
    </div>

    </body>
    </html>

    <?php $database->close_connection(); ?>
```

Data Providers

Checklist_Renderer

```
    <!-- Page Content -->
    <header class="custom-header">
        <h4><?php echo $current_list->name(); ?></h4>
    </header>
    <div class="container-fluid">
        <!-- Heading Row -->
        <div class="panel-group" id="accordion">
            <?php
            $tasks =
    Task::find_all_for_checklist($current_list->id);
            while ($task = current($tasks)) {
                $task->init_members();
                $user = $task->get_user();
                ?>
```

```
            <div class="panel panel-<?= $task-
>get_class(); ?>">
                <div class="panel-heading">
                    <a data-toggle="collapse" data-
parent="#accordion" href="#task-<?php echo $task->id; ?>">
                        <h6 class="panel-title">
                            <?php echo $task-
>avatar("48px"); ?>
                            <?php echo $task->name(); ?>
                        </h6>
                    </a>
                </div>
                <div id="task-<?php echo $task->id; ?>"
class="panel-collapse collapse out">
                    <div class="panel-body">
                        <blockquote>
                            <?php echo $task->details; ?>
                        </blockquote>
                        <p>
                            Due by: <?php echo $task-
>deadline(); ?>
                        </p>
                    </div>
                    <div class="panel-footer">
                        <?php if ($task->assigned_to ==
$current_user->id || $current_event->organiser ==
$current_user->id): ?>

                            <?php if (!$current_event-
>can_be_rated()) : ?>

                                <select name="task"
class="form-control" onchange="modify_task(<?php echo
$task->id; ?>, this.value);return false;">
                                    <?php
                                    $selected_option =
$task->status;
                                    $options =
array('Assigned', 'Started', 'Working', 'Completed',
'Failed');
                                    include
'./layouts/data/options_list.php';
                                    ?>
                                </select>
                                <span id="span-task-<?php
echo $task->id; ?>" class="text-success"></span>
                            <?php else: ?>
                                <?php echo $task->status;
?>
                            <?php endif; ?>
                        <?php else: ?>
                            <?php echo $task->status; ?>
                        <?php endif; ?>
```

```
                        <?php echo $user->avatar("32px") .
" " . $user->name(); ?>
                    </div>
                </div>
            </div>
            <?php
            next($tasks);
        }
        ?>
    </div>
    <?php
    if (!$current_event->can_be_rated()) :
        if ($current_user->id == $current_event-
>organiser) :
            ?>
            <div class="panel panel-default col-md-12">
                <h3 class="panel-heading text-
capitalize">Add a new Task</h3>
                <?php $object = new Task(); ?>

                <form id="form" class="panel-body"
method="post" action="tableForms/insert.php"
enctype="multipart/form-data">
                    <div class="form-group">
                        <label class="col-form-label"
for="title">Title</label>
                        <input id="title" name="title"
class="form-control" type="text" value="<?php echo
$object->title; ?>" required />
                    </div>

                    <div class="form-group">
                        <label class="col-form-label"
for="assigned_to">Assign To</label>
                        <select id="assigned_to"
name="assigned_to" class="form-control" required>
                            <?php
                            $options =
User::find_members_of_event($current_event->id);
                            include 'options_list.php';
                            ?>
                        </select>
                    </div>

                    <div class="form-group">
                        <label class="col-form-label"
for="deadline">Deadline</label>
                        <input id="deadline"
name="deadline" class="form-control" type="<?php echo
($object->deadline) ? 'text' : 'datetime-local'; ?>"
value="<?php echo $object->deadline; ?>" required/>
                    </div>
                    <div class="form-group">
```

```
                        <label class="col-form-label"
for="img">Image</label>
                        <input id="img" name="img"
class="form-control" type="file" accept="image/*"  <?php
echo ($object->img) ? '' : 'required'; ?>/>
                    </div>


                    <div class="form-group">
                        <label class="col-form-label"
for="status">Status</label>
                        <select id="status" name="status"
class="form-control">
                            <?php
                            $selected_status = $object-
>status;
                            $options_rate =
array('Assigned', 'Started', 'Working', 'Completed',
'Failed');
                            foreach ($options_rate as
$option):
                                ?>
                                <option value="<?php echo
$option; ?>"
                                        <?php echo
($option === $selected_status) ? 'selected' : ''; ?> >
                                            <?php echo
$option; ?>
                                </option>
                            <?php endforeach; ?>
                        </select>
                    </div>

                    <div class="form-group">
                        <label class="col-form-label"
for="details">Details</label>
                        <textarea id="details"
name="details" class="form-control" required><?php echo
$object->details; ?></textarea>
                    </div>
                    <div class="row btn-group-vertical
col-md-6 col-md-offset-3">
                        <input id="table_name"
name="table_name" type="hidden" value="task"/>
                        <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                        <input name="checklist"
type="hidden" value="<?php echo $current_list->id; ?>"/>
                        <input class="form-control btn
btn-primary" type="submit" value="Submit"/>
                    </div>
                </form>
            </div>
```

```
        <?php else: ?>
            <div class="panel panel-danger">
                <h3 class="panel-heading text-capitalize
text-danger">Only Organizers can 'Add a new Task'</h3>
                <p>
                    Come back later to see if the
organizer has added new tasks in this checklist.
                </p>
            </div>
        <?php
        endif;
    endif;
    ?>
</div>
<!-- /.row -->
```

Event_Details.php

```
<?php
if ($current_event):
    $event_organiser = $current_event->get_organiser();
    $m_pos_db = Guest::is_guest($current_user->id,
$current_event->id);
    $m_pos = array_shift($m_pos_db);
    $position = strtolower($m_pos->position);
    ?>
    <div class="panel with-nav-tabs panel-default">
        <div class="panel-heading">
            <ul class="nav nav-tabs">
                <?php
                if (!$current_event->can_be_rated()) :
                    if ($current_user->id ==
$current_event->organiser) :
                        ?>
                        <li class="right">
                            <a href="#event_edit" data-
toggle="tab"><span class="glyphicon glyphicon-
edit"></span></a>
                        </li>
                        <?php
                    endif;
                endif;
                ?>
                <li class="active">
                    <a href="#event_description" data-
toggle="tab">Event Details</a>
                </li>
                <li>
                    <a href="#event_guests" data-
toggle="tab">Guests</a>
                </li>
```

```
                <?php if ($position == 'admin' ||
$position == 'member' || $current_event->organiser ==
$current_user->id): ?>
                    <li>
                        <a href="#event_items" data-
toggle="tab">Items</a>
                    </li>
                <?php endif; ?>
                <li>
                    <a href="#event_lists" data-
toggle="tab">Lists</a>
                </li>
                <li>
                    <a href="#event_images" data-
toggle="tab">Images</a>
                </li>
                <li>
                    <a href="#event_venue" data-
toggle="tab">Venue</a>
                </li>
                <?php if ($current_event->can_be_rated()):
?>
                    <li>
                        <a href="#event_reviews" data-
toggle="tab">Review</a>
                    </li>
                <?php endif; ?>
            </ul>
        </div>
        <div class="panel-body">
            <div class="tab-content">
                <div class="tab-pane fade in active"
id="event_description">
                    <header class="row">
                        <div class="col-md-10">
                            <div class="col-md-12">
                                <div class="col-md-
4"><?php echo $current_event->avatar("128px", "img img-
thumbnail zoom-img"); ?></div>
                                <div class="col-md-8">
                                    <p class="text-
justify"><?php echo $current_event->description; ?></p>
                                    <span class="text-
info"><?php echo $current_event->datetime(); ?></span>
                                </div>
                            </div>
                        </div>
                    </header>
                    <?php if ($current_user->id ==
$current_event->organiser) : ?>
                        <div class="row">
                            <?php include
'./layouts/data/event_schedules.php'; ?>
                        </div>
```

```
                    <?php endif; ?>
                </div>
                <?php include
'./layouts/data/event_guests.php'; ?>
                <?php include
'./layouts/data/event_items.php'; ?>
                <?php include
'./layouts/data/event_lists.php'; ?>
                <?php include
'./layouts/data/event_images.php'; ?>
                <?php include
'./layouts/data/event_venue.php'; ?>
                <?php
                if ($current_event->can_be_rated()) {
                    include
'./layouts/data/event_review.php';
                }
                ?>
                <?php include
'./layouts/data/event_edit.php'; ?>
            </div>
            <div class="panel-footer">
                <h6><strong>Organized By</strong></h6>
                <?php echo $event_organiser-
>avatar("48px"); ?>
                <?php echo $event_organiser->name(); ?>
            </div>
        </div>
    </div>
<?php else: ?>

<?php endif; ?>
```

Event_Edit.php

```
<?php if ($current_user->id == $current_event->organiser)
: ?>
    <div class="tab-pane fade" id="event_edit">
        <?php $object = $current_event; ?>
        <div class="panel panel-default">
            <form id="new_event_form" class="panel-body"
method="post" action="tableForms/insert.php"
enctype="multipart/form-data">

                <div class="form-group col-md-3">
                    <label class="col-form-label"
for="name">Event name</label>
                    <input id="name" name="name"
class="form-control" type="text" value="<?php echo
$object->name; ?>" required/>
                </div>
                <div class="form-group col-md-5">
                    <label class="col-form-label"
for="datetime">Date and Time</label>
```

```
                    <input id="datetime" name="datetime"
class="form-control" type="datetime-local" value="<?php
echo str_replace(" ", "T", $object->datetime); ?>"
required/>
                </div>
                <div class="form-group col-md-4">
                    <label class="col-form-label"
for="img">Image</label>
                    <input id="img" name="img"
class="form-control" type="file" accept="image/*" <?php
echo ($object->img) ? '' : 'required'; ?>/>
                </div>
                <div class="form-group col-md-12">
                    <label class="col-form-label"
for="description">Event Description</label>
                    <textarea id="description" rows="8"
name="description" class="form-control" required><?php
echo $object->description; ?></textarea>
                </div>
                <br>

                <div class="row btn-group-vertical col-md-
6 col-md-offset-3">
                    <input name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
                    <input id="id" name="id" class="form-
control" type="hidden" readonly value="<?php echo $object-
>id; ?>"/>
                    <input id="organiser" name="organiser"
type="hidden" value="<?php echo $current_user->id; ?>">
                    <input id="table_name"
name="table_name" type="hidden" value="event"/>
                    <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                    <input class="form-control btn "
type="reset" value="Clear"/>
                </div>

            </form>
        </div>
    </div>
<?php endif; ?>
```

Event_Guests.php

```
<div class="tab-pane fade active" id="event_guests">
    <ul class="list-group">
        <?php
        $table_records =
Guest::find_all_for_event($current_event->id);
        if ($table_records) :
            while ($guestO = current($table_records)):
```

```
                $guest = $guestO->get_user();
                ?>
                <li class="list-group-item list-group-
item-<?php echo $guestO->css_class(); ?>">
                    <?php echo $guest->avatar("72px", "img
img-thumbnail zoom-img", "-"); ?>
                    <strong><?php echo $guest->name();
?></strong>
                    <em>
                        (<?php echo $guestO->position; ?>)
                    </em>
                    <br>
                    <span
class="row"><strong>Status</strong> : <?php echo $guestO-
>status; ?></span>
                </li>
                <?php
                next($table_records);
            endwhile;
            ?>
        <?php else: ?>
            <p>
                Nobody has been invited to this event yet.
            </p>
        <?php endif; ?>
    </ul>
    <?php
    if (!$current_event->can_be_rated()) :
        if ($current_user->id == $current_event-
>organiser) :
            $object = new Invitation();
            ?>
            <div class="panel panel-default col-md-10 col-
md-offset-1">
                <h4 class="panel-heading text-
capitalize">Invite a new Guest</h4>
                <form id="form" class="panel-body"
method="post" action="./logic/invite_guests.php"
enctype="multipart/form-data">

                    <div class="form-group col-md-6">
                        <label class="col-form-label"
for="user">Guests</label>
                        <select name="users[]"
class="form-control" required multiple>
                            <?php
                            $options =
User::find_uninvited($current_event->id);
                            include
'./layouts/data/options_list.php';
                            ?>
                        </select>
                    </div>
                    <div class="form-group col-md-6">
```

```
                        <label class="col-form-label"
for="position">Position</label>
                        <select id="guest_select"
name="position" class="form-control">
                            <?php
                            $options_guest = array('Guest
of Honor', 'V.I.P', 'Guest', 'Member', 'Admin');
                            foreach ($options_guest as
$option):
                                ?>
                                <option value="<?php echo
$option; ?>">
                                    <?php echo $option; ?>
                                </option>
                            <?php endforeach; ?>
                        </select>
                    </div>
                    <div class="form-group col-md-12">
                        <label class="col-form-label"
for="message">Message</label>
                        <textarea id="message"
name="message" class="form-control" required><?php echo
$object->message; ?></textarea>
                    </div>
                    <div class="row btn-group-vertical
col-md-6 col-md-offset-3">
                        <input id="table_name"
name="table_name" type="hidden" value="invitation"/>
                        <input name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
                        <input id="event" name="event"
type="hidden" value="<?php echo $current_event->id; ?>"/>
                        <input id="status" name="status"
type="hidden" value="May Be"/>
                        <input class="form-control btn
btn-primary" type="submit" value="Submit"/>
                        <input class="form-control btn "
type="reset" value="Clear"/>
                    </div>
                </form>
            </div>
            <?php
        endif;
    endif;
    ?>
</div>
```

Event_Images.php

```
<div class="tab-pane fade" id="event_images">

    <div class="row">
```

```
        <?php
        $images =
Gallery::find_all_for_event($current_event->id);
        $imgCount = 0;
        while ($image = current($images)):
            ?>
            <div class="col-xs-12 col-md-6">
                <div class="panel panel-default">
                    <!--                <div class="panel-
heading">
                                        <h3 class="panel-
title">You can even have a Panel Title</h3>
                                    </div>-->
                    <div class="panel-image">
                        <a
href="./view_image.php?image_id=<?php echo $image->id; ?>"
target="_blank">
                            <img src="<?php echo $image-
>image_dir() . DS . $image->img; ?>" class="panel-image-
preview" />
                        </a>
                    </div>
                    <div class = "panel-body">
                        <p>
                            <?php echo $image->note;
                            ?>
                        </p>
                    </div>
                </div>
            </div>
            <?php
            $imgCount++;
            next($images);
        endwhile;
        if ($imgCount < 1) :
            ?>
            <h2 class="panel-heading ">Sorry no images for
this event yet.</h2>
            <?php
        endif;
        ?>

    </div>

    <?php
    if ($current_user->id == $current_event->organiser) :
        $object = new Gallery();
        ?>

        <div class="panel panel-default row">
            <h3 class="panel-heading text-
capitalize">Insert new Image</h3>
```

```
            <form id="form" class="panel-body form-inline"
method="post" action="tableForms/insert.php"
enctype="multipart/form-data">

                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" required/>

                <label class="col-form-label"
for="note">Note</label>
                <input id="note" name="note" class="form-
control" value="<?php echo $object->note; ?>"/>

                <input id="table_name" name="table_name"
type="hidden" value="gallery"/>
                <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>

                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
            </form>
        </div>

        <script>
            var formRules = {
                rules: {

                },
                messages: {

                }
            };
        </script>
    <?php endif; ?>

</div>
```

Event_Items.php

```
<?php
if ($position == 'admin' || $position == 'member' ||
$current_event->organiser == $current_user->id):
    ?>

    <div class="tab-pane fade active" id="event_items">
        <ul class="list-group">
            <?php
            $table_records = $current_event->get_items();
            if ($table_records) :
```

```
                while ($eventItem =
current($table_records)):
                    ?>
                    <li class="list-group-item">
                        <small>#<?php echo $eventItem->id;
?></small><br>
                        <?php echo $eventItem-
>avatar("48px", "img img-thumbnail zoom-img"); ?>
                        <em><?php echo $eventItem->type;
?></em>
                        <strong><?php echo $eventItem-
>name(); ?></strong>
                        <blockquote>
                            <?php echo $eventItem-
>note($current_event->id); ?>
                        </blockquote>
                    </li>
                    <?php
                    next($table_records);
                endwhile;
            else:
                ?>
                <p>
                    No item has yet been selected.
                </p>
            <?php endif; ?>
        </ul>

        <?php
        if (!$current_event->can_be_rated()) :
            if ($current_user->id == $current_event-
>organiser) :
                $object = new EventItem();
                ?>

                <div class="panel panel-default row">
                    <h3 class="panel-heading text-
capitalize">Add new Item</h3>
                    <form id="form" class="panel-body
form-inline" method="post" action="tableForms/insert.php"
enctype="multipart/form-data">
                        <label class="col-form-label"
for="item_type">Filter by Items Type</label>
                        <select class="form-control"
onchange="fetch_items_of_type(<?php echo $current_event-
>id; ?>, this.value);return false;">
                            <?php
                            $options =
Item::get_available_types();
                            include
'./layouts/data/options_list.php';
                            ?>
                            <option value="-"
selected>All</option>
```

```
                        </select>

                        <label class="col-form-label"
for="item">Item</label>
                        <select id="item" name="item"
class="form-control" required>
                            <?php
                            $options = $current_event-
>find_remaining_items();
                            include
'./layouts/data/options_list.php';
                            ?>
                        </select>
                        <label class="col-form-label"
for="note">Note</label>
                        <input type="text" id="note"
name="note" class="form-control" value="<?php echo
$object->note; ?>"/>

                        <input id="table_name"
name="table_name" type="hidden" value="eventitem"/>
                        <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>

                        <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                        <input class="form-control btn
btn-primary" type="submit" value="Add"/>
                    </form>
                </div>

                <script>
                    var formRules = {
                        rules: {

                        },
                        messages: {

                        }
                    };
                </script>
                <?php
            endif;
        endif;
        ?>

    </div>

<?php else: ?>
    <div class="tab-pane fade" id="event_lists">
        <h1 class="panel-heading text-danger">
            Restricted!!
        </h1>
```

```
        <p class="panel-body">
            "You are a <strong class="text-
capitalize"><?php echo $position; ?> </strong> and only
Organizer or Members of event are allowed to see
checklists."
        </p>
    </div>
<?php endif; ?>
```

Event_Lists.php

```
<?php
if ($position == 'admin' || $position == 'member' ||
$current_event->organiser == $current_user->id):
    ?>

    <div class="tab-pane fade" id="event_lists">
        <ul class="list-group">
            <?php
            $table_records =
CheckList::find_all_for_event($current_event->id);
            if ($table_records) :
                while ($list = current($table_records)):
                    ?>
                    <li class="list-group-item">
                        <a
onclick="showPopup(this.href);return(false);"
                           class="btn"

href="./view_checklist.php?list=<?php echo $list->id; ?>">
                               <?php echo $list-
>avatar("48px"); ?>
                            <strong><?php echo $list-
>name(); ?></strong>
                        </a>
                    </li>
                    <?php
                    next($table_records);
                endwhile;
            else:
                ?>
                <p>
                    No list has yet been created.
                </p>
            <?php endif; ?>
        </ul>

        <?php
        if (!$current_event->can_be_rated()) :

            if ($current_user->id == $current_event-
>organiser) :
                $object = new CheckList();
```

```
                ?>

                <div class="panel panel-default row">
                    <h3 class="panel-heading text-
capitalize">New Checklist</h3>
                    <form id="form" class="panel-body
form-inline" method="post" action="tableForms/insert.php"
enctype="multipart/form-data">
                        <label class="col-form-label"
for="title">Title</label>
                        <input id="title" name="title"
class="form-control" type="text" required/>
                        <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                        <label class="col-form-label"
for="img">Image</label>
                        <input id="img" name="img"
class="form-control" type="file" accept="image/*"
required/>
                        <input id="created_on"
name="created_on" class="form-control" type="hidden"
value="<?php echo date("Y-m-d") . 'T' . date("h:i:s"); ?>"
/>
                        <input id="table_name"
name="table_name" type="hidden" value="checklist"/>
                        <input name="user" type="hidden"
value="<?php echo $current_user->id; ?>"/>
                        <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                        <input class="form-control btn
btn-primary" type="submit" value="Create New"/>
                    </form>
                </div>

                <script>
                    var formRules = {
                        rules: {

                        },
                        messages: {

                        }
                    };
                </script>
                <?php
            endif;
        endif;
        ?>


    </div>

<?php else: ?>
```

```
    <div class="tab-pane fade" id="event_lists">
        <h1 class="panel-heading text-danger">
            Restricted!!
        </h1>
        <p class="panel-body">
            "You are a <strong class="text-
capitalize"><?php echo $position; ?> </strong> and only
Organizer or Members of event are allowed to see
checklists."
        </p>
    </div>
<?php endif; ?>
```

Event_Renderer.php

```
<!-- Page Content -->
<div class="container-fluid">
    <!-- Heading Row -->
    <div class="col-md-2">
        <div>
            <h4 class="list-group-item-heading"><?php echo
$page_title; ?></h4>
            <ul class="list-group">
                <?php
                while ($event = current($events)) {
                    if (($current_event->id == $event-
>id)) {
                        ?>
                        <li class="list-group-item active
current">
                            <?php echo $event->name; ?>
                        </li>
                        <?php
                    } else {
                        ?>
                        <li class="list-group-item">
                            <a href="?event=<?php echo
$event->id; ?>">
                                <?php echo $event->name;
?>
                            </a>
                        </li>
                        <?php
                    }
                    next($events);
                }
                ?>
                <li class="list-group-item">
                    <a class="btn btn-success"
href="./new_event.php">
                        <span class="glyphicon glyphicon-
plus"></span> Create New event
                    </a>
```

```
                </li>
            </ul>
        </div>

        <div>
            <h4 class="list-group-item-heading">Upcoming
Events</h4>
            <ul class="list-group">
                <?php
                while ($invite =
current($upcoming_events)) {
                    $invite->init_members();
                    $event = $invite->eventObj;
                    if (($current_event->id == $event-
>id)) {
                        ?>
                        <li class="list-group-item active
current">
                            <?php echo $event->name; ?>
                        </li>
                        <?php
                    } else {
                        ?>
                        <li class="list-group-item">
                            <a href="?event=<?php echo
$event->id; ?>">
                                <?php echo $event->name;
?>
                            </a>
                        </li>
                        <?php
                    }
                    next($upcoming_events);
                }
                ?>
            </ul>
        </div>

        <div>
            <h4>Past Events</h4>
            <select id="past_events"
onchange="openEvent(this.value)">
                <option>Events that are in Past.</option>
                <?php
                while ($invite = current($invited_events))
{
                    $invite->init_members();
                    $event = $invite->eventObj;
                    if (($current_event->id == $event-
>id)) {
                        ?>
                        <option selected value="<?php echo
$event->id; ?>">
                            <?php echo $event->name; ?>
```

```
                        </option>
                        <?php
                    } else {
                        ?>
                        <option value="<?php echo $event-
>id; ?>">
                            <?php echo $event->name; ?>
                        </option>
                        <?php
                    }
                    next($invited_events);
                }
                ?>
            </select>

            <script>
                $("#past_events").select2();
            </script>

        </div>
    </div>
    <!-- /.col-md-4 -->
<!--    <div id="new_event" class="col-md-6 col-md-offset-
1 <?php echo ($current_event) ? "collapse" : ""; ?>">
    <?php // include './layouts/data/event_new.php'; ?>
    </div>-->

    <div id="contents" class="col-md-8">
        <?php
        if ($current_event) {
            include './layouts/data/event_details.php';
        } else {
            ?>
            <p class="alert alert-danger">
                <span class="glyphicon glyphicon-
flag"></span>
                <strong>Sorry</strong>, No event has been
selected.
            </p>
            <p class="alert">
                <span class="glyphicon glyphicon-info-
sign"></span>
                <em>Select</em> an <strong>event
</strong>to get it's details.
            </p>
            <p class="alert alert-success">
                <span class="glyphicon glyphicon-
plus"></span>
                You can also create a
                <a class="alert-link"
href="./new_event.php">
                    <em><strong>New Event</strong></em>
                </a>
            </p>
```

```
            <?php
        }
        ?>
    </div>

    <div id="my_assigned_tasks" class="col-md-2">
        <h4>Assigned Tasks</h4>
        <?php
        $sql = "select * from task where assigned_to =
$current_user->id"
                . " and"
                . " checklist in"
                . " (select id from checklist"
                . " where event in"
                . " (select id from event"
                . " where"
                . " datetime >= CURRENT_DATE()"
                . ")"
                . ")"
                . " order by deadline, status";
        $tasks = Task::find_by_sql($sql);
        ?>
        <?php
        while ($task = current($tasks)) :
            $task->init_members();
            ?>
            <div class="panel panel-<?php echo $task-
>get_class(); ?>">
                <div class="panel-heading">
                    <a class="unstyled" data-
toggle="collapse" data-parent="#accordion" href="#task-
<?php echo $task->id; ?>">
                        <span class="panel-title text-
right text-<?php echo $task->get_class(); ?>)">
                            <?php echo $task-
>avatar("32px", "img img-thumbnail zoom-img", "-"); ?>
                            <?php echo $task->name(); ?>
                        </span>

                        <br>
                        <span>
                            Due by: <?php echo $task-
>deadline(); ?>
                        </span>
                    </a>
                </div>
                <div id="task-<?php echo $task->id; ?>"
class="panel-collapse collapse out">
                    <div class="panel-body zoom-panel">

                        <blockquote class="task_details">
                            <?php echo $task->details; ?>
                        </blockquote>
```

```
                        <div class="details">
                            <?php
                            $taskParent = $task-
>checklistObj;
                            $taskEvent = $taskParent-
>eventObj;
                            ?>

                            <footer class="details">
                                <p>
                                    <small>
                                        Event:
                                    </small>
                                    <?php echo $taskEvent-
>anchor("48px", "no-img", "img img-thumbnail"); ?>
                                    <br>
                                    <small>
                                        Checklist:
                                    </small>
                                    <?php echo
$taskParent->anchor("48px", "no-img", "img img-
thumbnail"); ?>
                                    <?php // echo
$taskParent->name();   ?>

                                </p>
                            </footer>
                        </div>
                    </div>
                    <div class="panel-footer">
                        <?php if ($task->assigned_to ==
$current_user->id || $current_event->organiser ==
$current_user->id): ?>
                            <select name="task"
class="form-control" onchange="modify_task(<?php echo
$task->id; ?>, this.value);return false;">
                                <?php
                                $selected_option = $task-
>status;
                                $options =
array('Assigned', 'Started', 'Working', 'Completed',
'Failed');
                                include
'./layouts/data/options_list.php';
                                ?>
                            </select>
                            <span id="span-task-<?php echo
$task->id; ?>" class="text-success"></span>
                        <?php endif; ?>
                    </div>
                </div>
            </div>
            <?php
            next($tasks);
```

```
            endwhile; // Tasks
            ?>
        </div>

        <!-- /.col-md-8 -->
    </div>
    <!-- /.row -->
```

Event_Review.php

```
    <div class="tab-pane fade active" id="event_reviews">
        <div class="row">
            <div class="col-md-6 col-md-offset-3">
                <h4 class="text-info">Event Ratings</h4>
                <?php
                $ratings =
EventReview::find_numerically($current_event->id);
                $ratings_available = FALSE;
                while ($rating = current($ratings)) :
                    if (key($ratings) != "total") :
                        $number = ($rating /
$ratings['total']) * 100;
                        $lal = number_format($number, 2, '.',
'');

                        switch (key($ratings)) {
                            case 'Fantastic':
                            case 'Great':
                                $class = 'success';
                                break;
                            case 'Good':
                                $class = 'info';
                                break;
                            case 'Average':
                                $class = 'warning';
                                break;
                            case 'Below Average':
                            case 'Poor':
                                $class = 'danger';
                                break;
                            default :
                                $class = 'info';
                        }
                        $ratings_available = TRUE;
                        ?>
                        <strong><span class="glyphicon
glyphicon-star text-<?php echo $class; ?>"></span>  <?php
echo key($ratings); ?></strong>
                        <div class="progress">
                            <div class="progress-bar progress-
bar-<?php echo $class; ?>" role="progressbar" aria-
valuenow="<?php echo $lal; ?>"
                                 style="width: <?php echo
$lal; ?>%"
```

```
                                aria-valuemin="0" aria-
valuemax="100">
                                    <?php echo $lal; ?>
                            </div>
                        </div>
                        <?php
                    endif;
                    next($ratings);
                endwhile;
                if (!$ratings_available):
                    ?>
                    <p class="text-warning">
                        Sorry but no one has rated this event
yet!
                    </p>
                <?php endif; ?>
            </div>
        </div>

        <h4 class="text-info">Event Reviews by Guests</h4>

        <ul class="list-group">
            <?php
            $table_records =
EventReview::find_all_for_event($current_event->id);
            if ($table_records) :
                while ($review = current($table_records)):
                    $review->init_members();
                    ?>
                    <li class="list-group-item list-group-
item-warning">
                        <?php echo $review->avatar("128px",
"img img-thumbnail zoom-img", "-"); ?>

                        <strong class="list-group-item-
heading"><?php echo $review->name(); ?></strong>
                        <span class="text-right
right"><small>Rating</small> : <?php echo $review->rating;
?></span>

                        <blockquote class="list-group-item-
text">
                            <?php echo $review->description;
?>
                        </blockquote>

                        <footer class="right">
                            <em>Posted By:</em> <?php echo
$review->userObj->intro("72px", "", "img img-thumbnail
zoom-img", "-") . " " . $review->userObj->name(); ?>
                        </footer>
                    </li>
                    <?php
                    next($table_records);
```

```
            endwhile;
        else:
            ?>
            <p>
                No one reviewed the event yet.
            </p>
        <?php endif; ?>
    </ul>

    <?php
    if (Guest::is_guest($current_user->id, $current_event->id) && $current_event->can_be_rated()) :
        $object = EventReview::find_for_event_by_user($current_event->id, $current_user->id);
        if (!$object) {
            $object = new EventReview();
        }
        ?>

        <div class="panel panel-default">
            <h3 class="panel-heading text-capitalize">Post a review</h3>
            <form id="form" class="panel-body" method="post" action="tableForms/insert.php" enctype="multipart/form-data">
                <div class="form-group col-md-4">
                    <label class="col-form-label" for="title">Title</label>
                    <input id="title" name="title" class="form-control" type="text" value="<?php echo $object->name(); ?>" required/>
                </div>
                <div class="form-group col-md-4">
                    <label class="col-form-label" for="img">Image</label>
                    <input id="img" name="img" class="form-control" type="file" accept="image/*" <?php echo ($object->img) ? "" : "required"; ?>/>
                </div>

                <div class="form-group col-md-4">
                    <label class="col-form-label" for="rating">Rating</label>
                    <select id="rating" name="rating" class="form-control">
                        <?php
                        $options = array('Fantastic', 'Great', 'Good', 'Average', 'Below Average', 'Poor');
                        $selected_option = $object->rating;
                        include './layouts/data/options_list.php';
                        ?>
```

```
                    </select>
                </div>

                <div class="form-group col-md-12">
                    <label class="col-form-label"
for="title">Description</label>
                    <textarea name="description"
class="form-control" type="text" required><?php echo
$object->description; ?></textarea>
                </div>
                <div>
                    <input class="form-control btn btn-
primary" type="submit" value="Post"/>

                    <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                    <input name="posted_on" class="form-
control" type="hidden" value="<?php echo date("Y-m-d") .
'T' . date("h:i:s"); ?>" />
                    <input name="table_name" type="hidden"
value="eventreview"/>
                    <?php if ($object->id): ?>
                        <input name="id" class="form-
control" type="hidden" readonly value="<?php echo $object-
>id; ?>"/>
                    <?php endif; ?>
                    <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                    <input name="user" type="hidden"
value="<?php echo $current_user->id; ?>"/>
                </div>
            </form>
        </div>

        <script>
            var formRules = {
                rules: {

                },
                messages: {

                }
            };
        </script>
    <?php endif; ?>
</div>
```

Event_Schedules.php

```
<div class="panel lakshay">
    <h4>Event Schedule</h4>
    <div id="schedule-accordion" class="panel-group">
        <?php
```

```
        $schedules =
Schedule::find_all_for_event($current_event->id);
        while ($schedule = current($schedules)):
            ?>
            <div class="panel panel-default">
                <div class="panel-heading">
                    <a data-toggle="collapse" data-
parent="#schedule-accordion" href="#schedule-<?php echo
$schedule->id; ?>">
                        <span class="panel-title">
                            <?php echo $schedule->name();
?> (<em><small><?php echo $schedule->datetime();
?></small></em>)
                        </span>
                    </a>
                </div>
                <div id="schedule-<?php echo $schedule-
>id; ?>" class="panel-collapse collapse">
                    <div class="panel-body">
                        <?php echo $schedule->description;
?>
                    </div>
                </div>
            </div>
            <?php
            next($schedules);
        endwhile;
        ?>
    </div>
    <?php if (!$current_event->can_be_rated()) : ?>
        <div class="panel panel-default">
            <h5 class="panel-heading">Make another
schedule</h5>
            <?php $object = new Schedule(); ?>
            <form id="form" class="panel-body form-inline"
method="post" action="tableForms/insert.php"
enctype="multipart/form-data">
                <input name="title" class="form-control"
type="text" placeholder="Title" required value="<?php echo
$object->title; ?>"/>
                <input name="datetime" class="form-
control" type="datetime-local" value="<?php echo $object-
>datetime; ?>"  required/>
                <input type="text" name="description"
class="form-control" placeholder="Description" required/>
                <input id="table_name" name="table_name"
type="hidden" value="schedule"/>
                <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                <input name="redirect_url" type="hidden"
readonly value="<?php echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
            </form>
```

```
        </div>
    <?php endif; ?>
</div>
```

Event_Venue.php

```
<div class="tab-pane" id="event_venue">
    <div class="panel panel-default col-md-10 col-md-
offset-1">
        <?php $event_venue =
Venue::find_for_event($current_event->id); ?>

        <?php if (isset($event_venue->id)): ?>
            <h1 class="panel-heading">
                <?php echo $event_venue->avatar("72px",
"img img-responsive img-thumbnail zoom-img", "-"); ?>
                <?php echo $event_venue->name(); ?>
            </h1>
            <div class="panel-body">
                <p class="text-justify text-primary">
                    <?php echo $event_venue->description;
?>
                </p>
                <p class="text-justify text-info">
                    <strong>Address: </strong>
                    <?php echo $event_venue->address; ?>
                </p>
            </div>
            <?php
        elseif ($current_user->id == $current_event-
>organiser) :
            if (!$current_event->can_be_rated()) :
                $object = new Venue();
                ?>
                <h4 class="panel-heading text-
capitalize">book venue</h4>
                <form id="form" class="panel-body"
method="post" action="tableForms/insert.php"
enctype="multipart/form-data">
                    <div class="row form-group">
                        <?php
                        $venues =
Venue::find_available($current_event->datetime("Y-m-d"));
                        while ($venue = current($venues)):
                            ?>
                            <div class="radio">
                                <label>
                                    <input type="radio"
name="venue" value="<?php echo $venue->id; ?>"/>
                                    <div>
                                        <div class="col-
md-2">
```

```
                                            <?php echo
$venue->avatar("128px", "img img-thumbnail img-responsive
zoom-img"); ?>
                                            <strong><?php
echo $venue->name(); ?></strong>
                                        </div>
                                        <div class="col-
md-8 col-md-offset-2">

<strong>Description</strong>
                                            <p>
                                                <?php echo
$venue->description; ?>
                                            </p>
                                            <p>

<strong>Capacity</strong>: <?php echo $venue->capacity; ?>
                                            </p>

<strong>Address</strong>
                                            <p>
                                                <?php echo
$venue->address; ?>
                                            </p>
                                        </div>
                                    </div>
                                </label>
                            </div>
                            <?php
                            next($venues);
                        endwhile;
                        ?>
                    </div>
                    <div class="row btn-group-vertical
col-md-6 col-md-offset-3">
                        <input name="bookedOn"
class="form-control" type="hidden"  value="<?php echo
date("Y-m-d") . 'T' . date("h:i:s"); ?>" />
                        <input id="table_name"
name="table_name" type="hidden" value="eventVenue"/>
                        <input name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
                        <input name="bookedBy"
type="hidden" value="<?php echo $current_user->id; ?>"/>
                        <input name="event" type="hidden"
value="<?php echo $current_event->id; ?>"/>
                        <input class="form-control btn
btn-primary" type="submit" value="Submit"/>
                        <input class="form-control btn "
type="reset" value="Clear"/>
                    </div>
                </form>
                <?php
```

```
                endif;
            else:
                ?>
                <h4>No Venue has been decided yet.</h4>

            <?php endif; ?>
        </div>
    </div>
```

Options_List.php

```
    <?php
    $selected_option = (isset($selected_option)) ?
    $selected_option : 0;
    while ($option = current($options)):
        if ($option instanceof DatabaseObject) {
            ?>
            <option title="<?php echo $option->title(); ?>"
    value="<?php echo $option->id; ?>"  <?php echo
    ($selected_option == $option->id) ? 'selected' : ''; ?> >
                <?php echo $option->name(); ?>
            </option>
        <?php } else {
            ?>
            <option title="<?php echo $option; ?>"
    value="<?php echo $option; ?>"  <?php echo
    ($selected_option == $option) ? 'selected' : ''; ?> >
                <?php echo $option; ?>
            </option>
            <?php
        }
        ?>
        <?php
        next($options);
    endwhile;
    ?>
```

Stylesheet

```
    /*@Color_1: #C33764;
    @Color_2: #1D2671;*/
    /*@Color_1 : lighten(@pink,48%);
    @Color_2: lighten(@blue,65%);*/
    /*@Color_1: #f2fde9;
    @Color_2 : #f8fde7;*/
    /*@color_extra_stroke: #489ab1;*/
    /*@Color_1: #f7fdf8;
    @Color_2: #fcffe5;*/
    body {
      /*background: @white;*/
      /*background: @Color_1;*/
      background: linear-gradient(to right, rgba(255, 248,
    226, 0.49), rgba(252, 253, 231, 0.43));
    }
    header nav {
      margin: 0px !important;
```

```
}
.site_logo.active .fill {
  fill: #44a3d6;
  stroke: #137692;
  -webkit-transition: all 500ms;
  transition: all 500ms;
}
.site_logo .fill {
  stroke: #44a3d6;
}
header.custom-header {
  padding: 12px;
  padding-left: 6px;
  background: #000000;
  color: #ffffff;
  margin-bottom: 16px;
}
.affix {
  background: #ffffff;
  border-radius: 4px;
  box-shadow: 0px 4px 16px 0px rgba(0, 0, 0, 0.22);
  top: 24px;
  z-index: 1000;
}
.error {
  color: #ff0000;
}
.current {
  background: #f3f3f3;
}
.current a {
  cursor: none;
}
.list-group-item.active {
  cursor: none;
}
.img-square {
  max-width: fit-content;
  object-fit: scale-down;
}
.img-square .img-256 {
  width: 256px;
  height: 256px;
}
.img-square .img-128 {
  width: 128px;
  height: 128px;
}
.img-square .img-96 {
  width: 96px;
  height: 96px;
}
.img-square .img-72 {
  width: 72px;
```

```
  height: 72px;
}
.img-square .img-48 {
  width: 48px;
  height: 48px;
}
.img-square .img-32 {
  width: 32px;
  height: 32px;
}
.zoom-img:hover {
  z-index: 1000;
  transform: scale(2.5);
  position: relative;
  left: 24px;
  transition-property: all;
  transition-delay: 100ms;
  transition-timing-function: cubic-bezier(0.49, 0.01,
0.48, 1.43);
}
.zoom-img.stay:hover {
  left: 0;
}
.zoom-panel:hover {
  transition: 100ms;
  padding: 24px;
  z-index: 1000;
  background: #ffffff;
  position: relative;
  top: 0px;
  height: 300px;
  left: -100px;
  width: 300px;
  -webkit-box-shadow: 16px -4px 64px 12px rgba(0, 0, 0,
0.22);
  -moz-box-shadow: 16px -4px 64px 12px rgba(0, 0, 0,
0.22);
  box-shadow: 16px -4px 64px 12px rgba(0, 0, 0, 0.22);
  border-bottom: 1px solid rgba(0, 0, 0, 0.44);
  border-radius: 4px;
}
.zoom-panel:hover div.details {
  display: block;
}
.zoom-panel:hover blockquote.task_details {
  overflow: visible;
  overflow-x: auto;
  height: 150px;
}
.zoom-panel div.details {
  display: none;
}
.zoom-panel blockquote.task_details {
  height: 100px;
```

```
  overflow: hidden;
}
.zoom-panle .img.img-thumbnail {
  display: none;
}
a.no-img img {
  display: none;
}
a.no-img:hover img {
  display: block;
  position: absolute;
  bottom: 55px;
  left: 40px;
  box-shadow: 0px 0px 4px 4px rgba(0, 0, 0, 0.22);
}
footer.details {
  position: absolute;
  bottom: 0px;
}
.container-fluid {
  padding-top: 24px;
}
.lakshay {
  padding: 24px;
}
a.unstyled:hover {
  text-decoration: none !important;
}
a.unstyled:focus,
a.unstyled:active {
  font-style: italic;
  text-decoration: none !important;
}
.panel-image img.panel-image-preview {
  width: 100%;
  border-radius: 4px 4px 0px 0px;
}
.panel-image ~ .panel-body {
  overflow: hidden;
}
.panel-image ~ .panel-footer a {
  padding: 0px 12px;
  font-size: 1.3em;
  color: #646464;
}
.panel-heading ~ .panel-image img.panel-image-preview {
  border-radius: 0px;
}
.panel-image.hide-panel-body ~ .panel-body {
  overflow: hidden;
  height: 0px;
  padding: 0px;
}
#photograph {
```

```
  border: 8px solid #ffffff;
  margin-left: auto;
  margin-right: auto;
  margin-bottom: 24px;
  width: 90%;
  height: 90%;
  object-fit: scale-down;
  box-shadow: 4px 4px 4px 1px rgba(0, 0, 0, 0.15);
}
#photograph + blockquote {
  position: static;
  left: 0px;
  bottom: 0px;
  padding: 12px;
  margin-left: 12px;
  margin-bottom: 64px;
  background: white;
  border-left: 4px solid rgba(255, 199, 22, 0.49);
  width: 50%;
  box-shadow: 0px 0px 4px 1px rgba(0, 0, 0, 0.22);
  border-top-right-radius: 4px;
  border-bottom-right-radius: 4px;
  z-index: 1000;
}
aside#img_comments {
  min-height: 550px;
  max-height: 80%;
  box-shadow: -4px 4px 4px 0px rgba(0, 0, 0, 0.15);
}
aside#img_comments section {
  min-height: 50%;
  max-height: 550px;
  overflow: overlay;
}
aside#img_comments section ul {
  margin-right: 12px;
  margin-bottom: 64px;
}
aside#img_comments form {
  padding-bottom: 8px;
  position: absolute;
  bottom: 0;
  margin-top: 16px;
  margin-bottom: 16px;
}
::-webkit-scrollbar {
  background: #101010;
  border-radius: 8px;
  width: 8px;
}
::-webkit-scrollbar-thumb {
  /*background: darken(@Color_1, 40%);*/
  background: #74c6f3;
  border-radius: 8px;
```

```
  /*-webkit-box-shadow: inset 0 0 2px goldenrod;*/
}
#page_footer {
  background: #101010;
  color: #f9f9f9;
  padding: 16px;
  border-top: 4px solid rgba(255, 199, 22, 0.49);
}
#select_tables {
  background: #ffffff;
  border: 1px solid rgba(226, 171, 0, 0.49);
}
#to_record {
  float: right;
  background: #ffffff;
  padding: 16px;
  border: 1px solid;
  box-shadow: 0px 0px 8px 0px rgba(0, 0, 0, 0.15);
  border-radius: 8px;
}
#to_record:after {
  content: "";
  clear: both;
}
/*# sourceMappingURL=lakshay.css.map */
```

JavaScript

```
function showPopup(url) {
    newwindow = window.open(url, "Checklist",
'height=480,width=320,top=50,left=200,location=0,resizable
');
    if (window.focus) {
        newwindow.focus()
    }
}

function openEvent(event) {
    if (event > 0) {
        window.location =
"http://localhost/events/view_event.php?event=" + event;
    }
}

function modify_task(task_id, status) {
    $.post("./logic/modify_task.php",
            {
                task_id: task_id,
                status: status
            },
            function (response) {
//                alert("Updated :" + response.title + "
to " + response.status);
                $("#span-task-" +
task_id).html("Updated");
            }, 'json'
            );

}


var notificationsCount = 0;
function attend_event(invitation_id, status) {
    $.post("./logic/attend.php",
            {
                invitation_id: invitation_id,
                status: status
            },
            function (response) {
                $('#invite-' + invitation_id + "
div.response").html(response.msg);
            }, 'json'
            );
    notificationsCount--;
    updateNotificationsCount();
}
function notifications(nf) {
    notificationsCount = nf;
    updateNotificationsCount();
}
```

```
function updateNotificationsCount() {
    $("#notificationsCount").html(notificationsCount);
}
function fetch_items_of_type(event, type) {
    $.post("./logic/item_options.php",
            {
                type: type,
                event: event
            },
            function (response) {
                $('#item').html(response);
                return response;
            }
    );
}
```

Data Insertion Forms

Checklist Form

```
<?php
if (isset($_GET['id'])) {
    $object = CheckList::find_by_id($_GET['id']);
} else {
    $object = new CheckList();
}
?>

<div class="panel panel-default col-md-8 col-md-offset-2">
    <h3 class="panel-heading text-capitalize">Insert new
Checklist</h3>
    <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
        <?php if ($object->id): ?>
            <div class="form-group col-md-2">
                <label class="col-form-label"
for="id">Id</label>
                <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
            </div>
        <?php endif; ?>
        <div class="form-group col-md-4">
            <label class="col-form-label"
for="title">Title</label>
            <input id="title" name="title" class="form-
control" type="text" value="<?php echo $object->title; ?>"
required/>
        </div>
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="img">Image</label>
```

```
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" <?php if (!$object-
>id) echo 'required'; ?>/>
            </div>


        <div class="form-group col-md-6">
            <label class="col-form-label"
for="user">Created BY</label>
            <select id="user" name="user" class="form-
control" required>
                <?php
                $selected_user = $object->user;
                include 'userSelect.php';
                ?>
            </select>
        </div>
        <div class="form-group col-md-6">
            <label class="col-form-label"
for="event">Select an Event</label>
            <select id="event" name="event" class="form-
control" required>
                <?php
                $selected_event = $object->event;
                include 'eventSelect.php';
                ?>
            </select>
        </div>
        <div class="row btn-group-vertical col-md-6 col-
md-offset-3">
            <input id="table_name" name="table_name"
type="hidden" value="checklist"/>
            <input id="redirect_url" name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
            <input id="created_on" name="created_on"
class="form-control" type="hidden" value="<?php echo
form_date_time(); ?>" />
            <input class="form-control btn  btn-primary"
type="submit" value="Submit"/>
            <input class="form-control btn " type="reset"
value="Clear"/>
        </div>
    </form>
</div>

<script>
    var formRules = {
        rules: {

        },
        messages: {

        }
```

```
    };
</script>
```

Checklist Select

```
<?php
$selected_checklist = (isset($selected_checklist)) ?
$selected_checklist : 1;
$checkLists = CheckList::find_all();
while ($checklist = current($checkLists)):
    ?>
    <option value="<?php echo $checklist->id; ?>"  <?php
echo ($selected_checklist == $checklist->id) ? 'selected'
: ''; ?> >
        <?php echo $checklist->name(); ?>
    </option>

    <?php
    next($checkLists);
endwhile;
?>
```

Event Form

```
<?php
if (isset($_GET['id'])) {
    $object = Event::find_by_id($_GET['id']);
} else {
    $object = new Event();
}
?>

<div class="container-fluid">
    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new Event</h3>
        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>
            <div class="form-group col-md-5">
                <label class="col-form-label"
for="organiser">Organizer</label>
                <select id="organiser" name="organiser"
class="form-control" required>
                    <?php
                    $selected_user = $object->organiser;
```

```
                    include 'userSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-5">
                <label class="col-form-label"
for="name">Event name</label>
                <input id="name" name="name" class="form-
control" type="text" value="<?php echo $object->name; ?>"
required/>
            </div>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="datetime">Date and Time</label>
                <input id="datetime" name="datetime"
class="form-control" type="datetime-local" value="<?php
echo str_replace(" ", "T", $object->datetime); ?>"
required/>
            </div>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" <?php echo ($object-
>img) ? '' : 'required'; ?>/>
            </div>
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="description">Event Description</label>
                <textarea id="description"
name="description" class="form-control" required><?php
echo $object->description; ?></textarea>
            </div>
            <br>

            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="event"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>

        </form>
    </div>
</div>


<script>
```

```
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Event Item Form

```
<?php
if (isset($_GET['id'])) {
    $object = EventItem::find_by_id($_GET['id']);
} else {
    $object = new EventItem();
}
?>


<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Book
Item for event.</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">

            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="event">Event</label>
                <select id="event" name="event"
class="form-control" required>
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="item">Item</label>
```

```
                <select id="item" name="item" class="form-
control" required>
                    <?php
                    $selected_item = $object->item;
                    include 'itemSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="note">Note</label>
                <textarea id="note" name="note"
class="form-control"><?php echo $object->note;
?></textarea>
            </div>

            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="eventItem"/>
                <input id="bookedOn" name="bookedOn"
class="form-control" type="hidden"  value="<?php echo
date("Y-m-d") . 'T' . date("h:i:s"); ?>" />
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Event Review Form

```
<?php
if (isset($_GET['id'])) {
    $object = EventReview::find_by_id($_GET['id']);
} else {
    $object = new EventReview();
}
?>
```

```
<div class="panel panel-default col-md-8 col-md-offset-2">
    <h3 class="panel-heading text-capitalize">Insert new
Event Review</h3>

    <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">

        <?php if ($object->id): ?>
            <div class="form-group col-md-2">
                <label class="col-form-label"
for="id">Id</label>
                <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
            </div>
        <?php endif; ?>


        <div class="form-group col-md-4">
            <label class="col-form-label"
for="event">Event</label>
            <select id="event" name="event" class="form-
control" required>
                <?php
                $selected_event = $object->event;
                include 'eventSelect.php'
                ?>
            </select>
        </div>
        <div class="form-group col-md-6">
            <label class="col-form-label"
for="user">Posted By</label>
            <select id="user" name="user" class="form-
control" required>
                <?php
                $selected_user = $object->user;
                include 'userSelect.php';
                ?>
            </select>
        </div>
        <div class="form-group col-md-4">
            <label class="col-form-label"
for="title">Title</label>
            <input id="title" name="title" class="form-
control" type="text" value="<?php echo $object->title; ?>"
<?php echo ($object->title) ? '' : 'required'; ?>/>
        </div>

        <div class="form-group col-md-4">
            <label class="col-form-label"
for="img">Image</label>
```

```
            <input id="img" name="img" class="form-
control" type="file" accept="image/*"  <?php echo
($object->img) ? '' : 'required'; ?>/>
        </div>

        <div class="form-group col-md-4">
            <label class="col-form-label"
for="rating">Rating</label>
            <select id="rating" name="rating" class="form-
control">
                <?php
                $selected_rating = $object->rating;
                include 'ratingSelect.php';
                ?>
            </select>
        </div>
        <div class="form-group col-md-12">
            <label class="col-form-label"
for="description">Description</label>
            <textarea id="description" name="description"
class="form-control" placeholder="A few words about the
event" required><?php echo $object->description;
?></textarea>
        </div>
        <br>
        <div class="row btn-group-vertical col-md-6 col-
md-offset-3">
            <input id="table_name" name="table_name"
type="hidden" value="eventReview"/>
            <input id="posted_on" name="posted_on"
class="form-control" type="hidden" value="<?php echo
date("Y-m-d") . 'T' . date("h:i:s"); ?>" />
            <input id="redirect_url" name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
            <input class="form-control btn  btn-primary"
type="submit" value="Submit"/>
            <input class="form-control btn " type="reset"
value="Clear"/>
        </div>
    </form>
</div>


<script>
    var formRules = {
        rules: {
        },
        messages: {

        }
    };
</script>
```

Event Select

```
<?php
$selected_event = (isset($selected_event)) ?
$selected_event : 1;
$events = Event::find_all();
while ($event = current($events)):
    ?>
    <option value="<?php echo $event->id; ?>" <?php echo
($selected_event == $event->id) ? 'selected' : ''; ?>>
        <?php echo $event->name(); ?>
    </option>

    <?php
    next($events);
endwhile;
?>
```

Event Venue Form

```
<?php
if (isset($_GET['id'])) {
    $object = EventVenue::find_by_id($_GET['id']);
} else {
    $object = new EventVenue();
}
?>


<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Book a
Venue</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">

            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="event">Event</label>
                <select id="event" name="event"
class="form-control" required>
```

```
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="venue">Venue</label>
                <select id="venue" name="venue"
class="form-control" required>
                    <?php
                    $selected_venue = $object->venue;
                    include 'venueSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="bookedBy">Booked By</label>
                <select id="bookedBy" name="bookedBy"
class="form-control" required>
                    <?php
                    $selected_user = $object->bookedBy;
                    include 'userSelect.php';
                    ?>
                </select>
            </div>
            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="eventVenue"/>
                <input id="bookedOn" name="bookedOn"
class="form-control" type="hidden"  value="<?php echo
date("Y-m-d") . 'T' . date("h:i:s"); ?>" />
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
```

```
</script>
```

Gallery Form

```
<?php
$tbClass = ucfirst($table);
if (isset($_GET['id'])) {
    $object = $tbClass::find_by_id($_GET['id']);
} else {
    $object = new $tbClass();
}
?>

<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new <?php echo $table; ?> Image</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">

            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="event">Select an Event</label>
                <select id="event" name="event"
class="form-control" required>
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php';
                    ?>
                </select>
            </div>

            <div class="form-group col-md-6">
                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" <?php if (!$object-
>id) echo 'required'; ?>/>
            </div>
```

```
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="note">Note</label>
                <textarea id="note" name="note"
class="form-control" rows="5"><?php echo $object->note;
?></textarea>
            </div>

            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="<?php echo $table; ?>"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Guest Form

```
<?php
if (isset($_GET['id'])) {
    $object = Guest::find_by_id($_GET['id']);
} else {
    $object = new Guest();
}
?>


<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Invite a
Guest</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php">
```

```
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-10">
                <label class="col-form-label"
for="user">Guest</label>
                <select id="user" name="user" class="form-
control" required>
                    <?php
                    $selected_user = $object->user;
                    include 'userSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="event">Event</label>

                <select id="event" name="event"
class="form-control" required>
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="position">Position</label>
                <select id="position" name="position"
class="form-control">
                    <?php
                    $options_guest = array('Guest of
Honor', 'V.I.P', 'Guest', 'Member', 'Admin');
                    foreach ($options_guest as $option):
                        ?>
                        <option value="<?php echo $option;
?>"
                                    <?php echo ($option ===
$object->position) ? 'selected' : ''; ?> >
                                        <?php echo $option; ?>
                        </option>
                    <?php endforeach; ?>
                </select>
            </div>
            <div class="form-group col-md-4">
```

```
                <label class="col-form-label"
for="status">Invitation Status</label>
                <select id="status" name="status"
class="form-control">
                    <?php
                    $options_invite = array('Attending',
'Not Attending', 'May Be');
                    foreach ($options_invite as $option):
                        ?>
                        <option value="<?php echo $option;
?>"
                                <?php echo ($option ===
$object->status) ? 'selected' : ''; ?> >
                                    <?php echo $option; ?>
                        </option>
                    <?php endforeach; ?>

                </select>
            </div>


            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="guest"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Image Comment Form

```
<?php
$tbClass = ucfirst($table);
if (isset($_GET['id'])) {
    $object = $tbClass::find_by_id($_GET['id']);
} else {
    $object = new $tbClass();
```

```
}
?>

<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new <?php echo $table; ?></h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-5">
                <label class="col-form-label"
for="image">Image</label>
                <select id="image" name="image"
class="form-control" required>
                    <?php
                    $options = Gallery::find_all();
                    $selected_option = $object->image;
                    include
'./layouts/data/options_list.php';
                    ?>
                </select>
            </div>

            <div class="form-group col-md-5">
                <label class="col-form-label"
for="user">Commented By</label>
                <select id="user" name="user" class="form-
control" required>
                    <?php
                    $selected_user = $object->user;
                    include 'userSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-5">
                <label class="col-form-label"
for="datetime">Comment On</label>
                <input id="datetime" name="datetime"
type="datetime-local" class="form-control" value="<?php
```

```
echo DatabaseObject::form_date($object->datetime); ?>"
required>
            </div>
            <div class="form-group col-md-7">
                <label class="col-form-label"
for="comment">Comment</label>
                <textarea id="comment" name="comment"
class="form-control" required><?php echo $object->comment;
?></textarea>
            </div>

            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="<?php echo $table; ?>"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Invitation Form

```
<?php
$tbClass = ucfirst($table);
if (isset($_GET['id'])) {
    $object = $tbClass::find_by_id($_GET['id']);
} else {
    $object = new $tbClass();
}
$object->init_members();
?>

<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new <?php echo $table; ?></h3>
```

```
        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-5">
                <label class="col-form-label"
for="user">Invite</label>
                <select id="user" name="user" class="form-
control" required>
                    <?php
                    $selected_user = $object->user;
                    include 'userSelect.php';
                    ?>
                </select>
            </div>
            <div class="form-group col-md-5">
                <label class="col-form-label"
for="event">Select an Event</label>
                <select id="event" name="event"
class="form-control" required>
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php';
                    ?>
                </select>
            </div>

            <div class="form-group col-md-12">
                <label class="col-form-label"
for="message">Message</label>
                <textarea id="message" name="message"
class="form-control" required><?php echo $object->message;
?></textarea>
            </div>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="position">Position</label>
                <select id="position" name="position"
class="form-control">
                    <?php
                    $options_guest = array('Guest of
Honor', 'V.I.P', 'Guest', 'Member', 'Admin');
                    foreach ($options_guest as $option):
```

```
                        ?>
                        <option value="<?php echo $option;
?>"
                                <?php echo ($option ===
$object->guestObj->position) ? 'selected' : ''; ?> >
                                    <?php echo $option; ?>
                        </option>
                    <?php endforeach; ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="status">Invitation Status</label>
                <select id="status" name="status"
class="form-control">
                    <?php
                    $options_invite = array('Attending',
'Not Attending', 'May Be');
                    foreach ($options_invite as $option):
                        ?>
                        <option value="<?php echo $option;
?>"
                                <?php echo ($option ===
$object->guestObj->status) ? 'selected' : ''; ?> >
                                    <?php echo $option; ?>
                        </option>
                    <?php endforeach; ?>

                </select>
            </div>



            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="<?php echo $table; ?>"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
```

```
        }
    };
</script>
```

Item Form

```
<?php
if (isset($_GET['id'])) {
    $object = Item::find_by_id($_GET['id']);
} else {
    $object = new Item();
}
?>

<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new Item</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>

            <div class="form-group col-md-6">
                <label class="col-form-label"
for="title">Title</label>
                <input id="title" name="title"
class="form-control" type="text"  required value="<?php
echo $object->title; ?>" />
            </div>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="type">Type</label>
                <input id="type" name="type" class="form-
control" type="text"  required value="<?php echo $object-
>type; ?>"/>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="img">Image</label>
```

```
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" <?php echo ($object-
>img) ? '' : 'required'; ?>/>
            </div>
            <div class="form-group col-md-8">
                <label class="col-form-label"
for="description">description</label>
                <textarea id="description"
name="description" class="form-control" required><?php
echo $object->description; ?></textarea>
            </div>

            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="item"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Item Review Form

```
<?php
if (isset($_GET['id'])) {
    $object = ItemReview::find_by_id($_GET['id']);
} else {
    $object = new ItemReview();
}
?>
<div class="panel panel-default col-md-8 col-md-offset-2">
    <h3 class="panel-heading text-capitalize">Insert new
Event Review</h3>

    <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
        <?php if ($object->id): ?>
```

```
            <div class="form-group col-md-2">
                <label class="col-form-label"
for="id">Id</label>
                <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
            </div>
        <?php endif; ?>

        <div class="form-group col-md-5">
            <label class="col-form-label"
for="item">Review For</label>
            <select id="item" name="item" class="form-
control" required>
                <?php
                $selected_item = $object->item;
                include 'itemSelect.php';
                ?>
            </select>
        </div>
        <div class="form-group col-md-5">
            <label class="col-form-label" for="event">In
Event</label>
            <select id="event" name="event" class="form-
control" required>
                <?php
                $selected_event = $object->event;
                include 'eventSelect.php'
                ?>
            </select>
        </div>
        <div class="form-group col-md-4">
            <label class="col-form-label"
for="title">Title</label>
            <input id="title" name="title" class="form-
control" type="text" value="<?php echo $object->title;
?>"/>
        </div>

        <div class="form-group col-md-4">
            <label class="col-form-label"
for="img">Image</label>
            <input id="img" name="img" class="form-
control" type="file" accept="image/*"  <?php echo
($object->img) ? '' : 'required'; ?>/>
        </div>
        <div class="form-group col-md-4">
            <label class="col-form-label"
for="rating">Rating</label>
            <select id="rating" name="rating" class="form-
control">
                <?php
                $selected_rating = $object->rating;
                include 'ratingSelect.php';
```

```
                ?>
            </select>
        </div>
        <div class="form-group col-md-12">
            <label class="col-form-label"
for="description">Description</label>
            <textarea id="description" name="description"
class="form-control" placeholder="A few words about the
event"><?php echo $object->description; ?></textarea>
        </div>

        <div class="form-group col-md-6">
            <label class="col-form-label"
for="user">Posted By</label>
            <select id="user" name="user" class="form-
control" required>
                <?php
                $selected_user = $object->user;
                include 'userSelect.php';
                ?>
            </select>
        </div>

        <br>
        <div class="row btn-group-vertical col-md-6 col-
md-offset-3">
            <input id="table_name" name="table_name"
type="hidden" value="itemReview"/>
            <input id="posted_on" name="posted_on"
class="form-control" type="hidden" value="<?php echo
date("Y-m-d") . 'T' . date("h:i:s"); ?>" />
            <input id="redirect_url" name="redirect_url"
type="hidden" readonly value="<?php echo
$_SERVER["REQUEST_URI"]; ?>"/>
            <input class="form-control btn  btn-primary"
type="submit" value="Submit"/>
            <input class="form-control btn " type="reset"
value="Clear"/>
        </div>
    </form>
</div>


<script>
    var formRules = {
        rules: {
        },
        messages: {

        }
    };
</script>
```

Item Select

```
<?php
$selected_item = (isset($selected_item)) ? $selected_item
: 1;
$items = Item::find_all();
while ($item = current($items)):
    ?>
    <option value="<?php echo $item->id; ?>" <?php echo
($selected_item == $item->id) ? 'selected' : ''; ?>>
        <?php echo $item->name(); ?>
    </option>

    <?php
    next($items);
endwhile;
?>
```

Rating Select

Schedule Form

```
<?php
if (isset($_GET['id'])) {
    $object = Schedule::find_by_id($_GET['id']);
} else {
    $object = new Schedule();
}
?>
<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Schedule
Task</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>
            <div class="form-group col-md-3">
                <label class="col-form-label"
for="event">Event</label>
                <select id="event" name="event"
class="form-control" required>
                    <?php
                    $selected_event = $object->event;
                    include 'eventSelect.php'
```

```
                    ?>
                </select>
            </div>
            <div class="form-group col-md-3">
                <label class="col-form-label"
for="title">Title</label>
                <input id="title" name="title"
class="form-control" type="text"  required value="<?php
echo $object->title; ?>"/>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="datetime">Date Time</label>
                <input id="datetime" name="datetime"
class="form-control" type="<?php echo ($object->datetime)
? 'text' : 'datetime-local'; ?>" value="<?php echo
$object->datetime; ?>"  required/>
            </div>
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="description">Description</label>
                <textarea id="description"
name="description" class="form-control" required><?php
echo $object->description; ?></textarea>
            </div>
            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="schedule"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Task From

```
<?php
if (isset($_GET['id'])) {
```

```
    $object = Task::find_by_id($_GET['id']);
} else {
    $object = new Task();
}
?>
<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Create A
Task</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="title">Title</label>
                <input id="title" name="title"
class="form-control" type="text" value="<?php echo
$object->title; ?>" required />
            </div>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="checklist">Checklist</label>
                <select id="checklist" name="checklist"
class="form-control" required>
                    <?php
                    $selected_checklist = $object-
>checklist;
                    include 'checklistSelect.php'
                    ?>
                </select>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="assigned_to">Assign To</label>
                <select id="assigned_to"
name="assigned_to" class="form-control" required>
                    <?php
                    $selected_user = $object->assigned_to;
                    include 'userSelect.php'
                    ?>
                </select>
```

```
            </div>

            <div class="form-group col-md-4">
                <label class="col-form-label"
for="deadline">Deadline</label>
                <input id="deadline" name="deadline"
class="form-control" type="<?php echo ($object->deadline)
? 'text' : 'datetime-local'; ?>" value="<?php echo
$object->deadline; ?>" required/>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*"  <?php echo
($object->img) ? '' : 'required'; ?>/>
            </div>


            <div class="form-group col-md-4">
                <label class="col-form-label"
for="status">Status</label>
                <select id="status" name="status"
class="form-control">
                    <?php
                    $selected_status = $object->status;
                    $options_rate = array('Assigned',
'Started', 'Working', 'Completed', 'Failed');
                    foreach ($options_rate as $option):
                        ?>
                        <option value="<?php echo $option;
?>"
                                <?php echo ($option ===
$selected_status) ? 'selected' : ''; ?> >
                                    <?php echo $option; ?>
                        </option>
                    <?php endforeach; ?>
                </select>
            </div>

            <div class="form-group col-md-12">
                <label class="col-form-label"
for="details">Details</label>
                <textarea id="details" name="details"
class="form-control" required><?php echo $object->details;
?></textarea>
            </div>
            <div class="row btn-group-vertical col-md-6
col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="task"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
```

```
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

User From

```
<?php
if (isset($_GET['id'])) {
    $object = User::find_by_id($_GET['id']);
} else {
    $object = new User();
}
?>
<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new User</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
            <?php endif; ?>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="user_name">User Name</label>
                <input id="user_name" name="user_name"
class="form-control" type="text" required value="<?php
echo $object->user_name; ?>"/>
            </div>
            <div class="form-group col-md-4">
```

```
                <label class="col-form-label"
for="email">Email</label>
                <input id="email" name="email"
class="form-control" type="email" required value="<?php
echo $object->email; ?>"/>
            </div>
            <div class="form-group  col-md-2">
                <label class="col-form-label"
for="type">User Type</label>
                <select id="type" name="type" class="form-
control" required>
                    <option value="admin">Admin</option>
                    <option value="user">User</option>
                    <option value="client">Client</option>
                </select>
            </div>
            <div class="form-group col-md-12">
                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*" <?php if (!$object-
>id) echo 'required'; ?>/>
            </div>

            <div class="form-group col-md-6">
                <label class="col-form-label"
for="first_name">First Name</label>
                <input id="first_name" name="first_name"
class="form-control" type="text" required value="<?php
echo $object->first_name; ?>"/>
            </div>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="last_name">Last Name</label>
                <input id="last_name" name="last_name"
class="form-control" type="text" required value="<?php
echo $object->last_name; ?>"/>
            </div>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="password">Password</label>
                <input id="password" name="password"
class="form-control" type="password" value="<?php echo
$object->password; ?>" />
            </div>
            <div class="form-group col-md-6">
                <label class="col-form-label"
for="password2">Repeat Password</label>
                <input id="password2" name="password2"
class="form-control" type="password" value="<?php echo
$object->password; ?>"/>
            </div>
            <div class="form-group col-md-12">
```

```
                <label class="col-form-label"
for="address">Address</label>
                <textarea id="address" name="address"
class="form-control" required><?php echo $object->address;
?></textarea>
            </div>
            <br>
            <div class="btn-group-vertical col-sm-12 col-
sm-offset-0 col-md-6 col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="user"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="User"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>

<script>
    var formRules = {
        rules: {
            first_name: "required",
            last_name: "required",
            user_name: {
                required: true,
                minlength: 6
            },
            password: {
                required: true,
                minlength: 6,
                maxlength: 10
            },
            password2: {
                required: true,
                minlength: 6,
                equalTo: "#password",
                maxlength: 10
            },
            email: {
                required: true,
                email: true
            }
        },
        messages: {
            user_name: {
                required: "You must provide a user name to
login in future.",
                minlength: "Make username long so it is
unique."
```

```
            },
            password2: {
                equalTo: "Passwords should match."
            }
        }
    };
</script>
```

User Select

```
<?php
$selected_user = (isset($selected_user)) ? $selected_user
: 0;
$users = User::find_all();
while ($user = current($users)):
    ?>
    <option value="<?php echo $user->id; ?>" <?php echo
($selected_user == $user->id) ? 'selected' : ''; ?> >
        <?php echo $user->name(); ?>
    </option>

    <?php
    next($users);
endwhile;
?>
```

Venue From

```
<?php
if (isset($_GET['id'])) {
    $object = Venue::find_by_id($_GET['id']);
} else {
    $object = new Venue();
}
?>

<div class="container-fluid">

    <div class="panel panel-default col-md-8 col-md-
offset-2">
        <h3 class="panel-heading text-capitalize">Insert
new Venue</h3>

        <form id="form" class="panel-body" method="post"
action="tableForms/insert.php" enctype="multipart/form-
data">
            <?php if ($object->id): ?>
                <div class="form-group col-md-2">
                    <label class="col-form-label"
for="id">Id</label>
                    <input id="id" name="id" class="form-
control" type="number" readonly value="<?php echo $object-
>id; ?>"/>
                </div>
```

```
            <?php endif; ?>

            <div class="form-group col-md-6">
                <label class="col-form-label"
for="name">Venue Name</label>
                <input id="name" name="name" class="form-
control" type="text" value="<?php echo $object->name; ?>"
required/>
            </div>
            <div class="form-group col-md-4">
                <label class="col-form-label"
for="img">Image</label>
                <input id="img" name="img" class="form-
control" type="file" accept="image/*"  <?php echo
($object->img) ? '' : 'required'; ?>/>
            </div>
            <div class="form-group col-md-2">
                <label class="col-form-label"
for="capacity">Capacity</label>
                <input id="capacity" name="capacity"
class="form-control" type="number" value="<?php echo
$object->capacity; ?>" required/>
            </div>

            <div class="form-group col-md-5">
                <label class="col-form-label"
for="address">Address</label>
                <textarea id="address" name="address"
class="form-control"><?php echo $object->address;
?></textarea>
            </div>
            <div class="form-group col-md-5">
                <label class="col-form-label"
for="description">Description</label>
                <textarea id="description"
name="description" class="form-control"><?php echo
$object->description; ?></textarea>
            </div>

            <div class="col-md-6 btn-group-vertical col-
md-6 col-md-offset-3">
                <input id="table_name" name="table_name"
type="hidden" value="venue"/>
                <input id="redirect_url"
name="redirect_url" type="hidden" readonly value="<?php
echo $_SERVER["REQUEST_URI"]; ?>"/>
                <input class="form-control btn  btn-
primary" type="submit" value="Submit"/>
                <input class="form-control btn "
type="reset" value="Clear"/>
            </div>
        </form>
    </div>
</div>
```

```
<script>
    var formRules = {
        rules: {
        },
        messages: {
        }
    };
</script>
```

Venue Select

```
<?php
$selected_venue = (isset($selected_venue)) ?
$selected_venue : 1;

$venues = Venue::find_all();
while ($venue = current($venues)):
    ?>
    <option value="<?php echo $venue->id; ?>" <?php echo
($selected_venue == $venue->id) ? 'selected' : ''; ?>>
        <?php echo $venue->name(); ?>
    </option>

    <?php
    next($venues);
endwhile;
?>
```

Options List

```
<?php
$selected_option = (isset($selected_option)) ?
$selected_option : 0;
while ($option = current($options)):
    if ($option instanceof DatabaseObject) {
        ?>
        <option title="<?php echo $option->title(); ?>"
value="<?php echo $option->id; ?>"  <?php echo
($selected_option == $option->id) ? 'selected' : ''; ?> >
            <?php echo $option->name(); ?>
        </option>
    <?php } else {
        ?>
        <option title="<?php echo $option; ?>"
value="<?php echo $option; ?>"  <?php echo
($selected_option == $option) ? 'selected' : ''; ?> >
            <?php echo $option; ?>
        </option>
        <?php
    }
    ?>
    <?php
    next($options);
```

```
endwhile;
?>
```

Data Insert

```
<?php include '../includes/initialize.php'; ?>
<?php
$table = $_POST["table_name"];
if (isset($_POST['id']) && !empty($_POST['id'])) {
    $object = $table::find_by_id($_POST['id']);
} else {
    $object = new $table;
}
$attrs = $object->attributes();

foreach ($_POST as $attribute => $value) {
    if ($attribute == "table_name" || $attribute == "id")
{
        continue;
    }
    if (array_key_exists($attribute, $attrs) &&
!empty($value)) {
        $object->$attribute = $value;
    } else if (property_exists($table, $attribute)) {
        $object->$attribute = $value;
    }
}
?>
<?php
if ($object->validate_attributes($object-
>insertion_attributes())) {
    if (!empty($_FILES['img']['name'])) {
        $object->img = $object-
>upload_img($_FILES['img']);
    }
    $object->save();
    if ($table == 'user' && !$session->get_user_object()
instanceof User) {
        $session->login($object);
//        redirect_to("../" . $session->request_uri());
//        exit();
    }

    $redirect_url = (isset($_POST['redirect_url'])) ?
$_POST['redirect_url'] : "../index.php";
    redirect_to($redirect_url);
} else {
    echo 'Errors';
}
?>
<pre>
    <?php print_r($_POST); ?>
</pre>
<pre>
```

```
    <?php print_r($object); ?>
</pre>
```

Comments and Description of Coding Segments

Index

Once the user inputs URL of our website they will be redirected to this webpage to have an idea of what the website is all about.

The page includes:

- Introduction to website
- Introduction to the developer
- Some technical information
- Contact information of the webmin to request for changes

Login

To gain access of secure data users must go through this page and sign in using correct credentials and once they do they may visit other pages and access the data in them.

Events

This page is where most of the time is spent by any user. As this is where they can view all the events and upon selecting an event, they get to see following details as well.

- Event Description
    - This portion show the information about event such as what the event is and when the event will be happening.
    - Also we can add **schedule** for the event here in this tab.
- Guest List
    - Anyone can view the event guests but admins can also invite others to the event as well.
    - Invitation can also be personalised for every guest.
- Items
    - Prebooked and pre tested items can be selected for the event from this page easily
- Checklists
-
- Venue
- Gallery
- Editing Event

Checkout

Once booking's stay is over, user can proceed to checkout from the hotel by confirming his billed amount and applying discounts on it if any. Once they do an invoice is generated for their filing purposes.

View Bill

Invoice generated during the checkout can be seen here and printed as well. The page is optimised to support printers and print only the required contents.

Admin List Tables

Admins of the site are allowed to view/edit/delete any record that is in the database to facilitate the hotel management and futher help the guests.

Also this page is able to generate reports for all the tables.

Database

This class facilitates the process of query database tables securely and easily after creating a conncetion with information provided in **Config.php** file.

Also this class handles code to get the record rows as an array from the result set we get after executing a query.

Insertion/Updation of row(s) also results in giving back the id of rows affected and their count as well.

Database Object

This class helps in conversion of record array into Table object to use them properly in the code. Also functions like *save(), update(), delete()* helps in performing the tasks more effectively.

Table Class

Every table has its own unique class to facilitate their personal tasks. Such as finding all the records, finding orders placed in a specific booking, generating invoice for a booking, etc.

Session

User login, logout, message handling and redirection tasks can not be done easily if we do not use sessions and session management is crucial for better code execution else it interferes with the actual code. That is why a separate class has been created for managing tasks mentioned before.

Standardization of the coding

Efforts have been made to write the entire code in a standard and formalised manner. For that purpose techniques like Object-Oriented programming, functions and methods, CSS for styling information and jQuery for client side experience are used.

PHP has the following conventions as well.

Naming Conventions

Global Variables and Functions

If your package needs to define global variables, their names should start with a single underscore followed by the package name and another underscore. For example, the PEAR package uses a global variable called $_PEAR_destructor_object_list.

Global functions should be named using the "studly caps" style (also referred to as "bumpy case" or "camel caps"). In addition, they should have the package name as a prefix, to avoid name collisions between packages. The initial letter of the name (after the prefix) is lowercase, and each letter that starts a new "word" is capitalized. An example:

XML_RPC_serializeData()

Classes

Classes should be given descriptive names. Avoid using abbreviations where possible. Class names should always begin with an uppercase letter. The PEAR class hierarchy is also reflected in the class name, each level of the hierarchy separated with a single underscore. Examples of good class names are:

Log	Net_Finger	HTML_Upload_Error

Class Variables and Methods

Class variables (a.k.a properties) and methods should be named using the "studly caps" style (also referred to as "bumpy case" or "camel caps"). Some examples (these would be "public" members):

$counter	connect()	getData()	buildSomeWidget()

Private class members are preceded by a single underscore. For example:

$_status	_sort()	_initTree()

Constants

Constants should always be all-uppercase, with underscores to separate words. Prefix constant names with the uppercased name of the class/package they are used in. Some examples:

DB_DATASOURCENAME	SERVICES_AMAZON_S3_LICENSEKEY

Code Efficiency

PHP's object orientation has been put to a good use in this project to make the code more and more efficient and light on the database as well.

Also using less as the css pre-processor has made the generation of stylesheet a lot easier.

jQuery helps in writing easy to read and tested code for client sides.

Error handling

PHP does a good job in handling the error prone code by itself and to ensure that the whole website is error free all the cases have been put inside if else blocks to test the values before performing any action.

Some examples are:

- Checking if Constant is already defined before creating it by calling defined function in php
- Using issset() to ensure that POST values from the Form were actually submitted
- Checking for null values before inserting them into database.
- Santizing the attributes and SQL calls to prevent faulty data.

Parameter Passing and Calling

Our website use two technologies PHP and JavaScript to create communication between a user and our system and following techniques are used to pass parameters in the website.

- **Client to Server:** Using Request parameters ie GET and POST.
    - This is sent using either a Form or JavaScript AJAX calls like $.ajax() and $.get(), $.post()
- **Server to Server:** If we are in same script this is done by function calls like find_by_id($id) and once we change scripts we need to send GET or POST request just like a client would.

By sending data by passing parameters we help in making our code more modular and easy to reuse as well.

Validation checks

HTML5 validates the data automatically if we instruct it in the form field using the tags like required, min, max, type.

But to make sure everything is tested before submission. We have used the jQuery validate data before submissions.

Screen Shots

Index.php

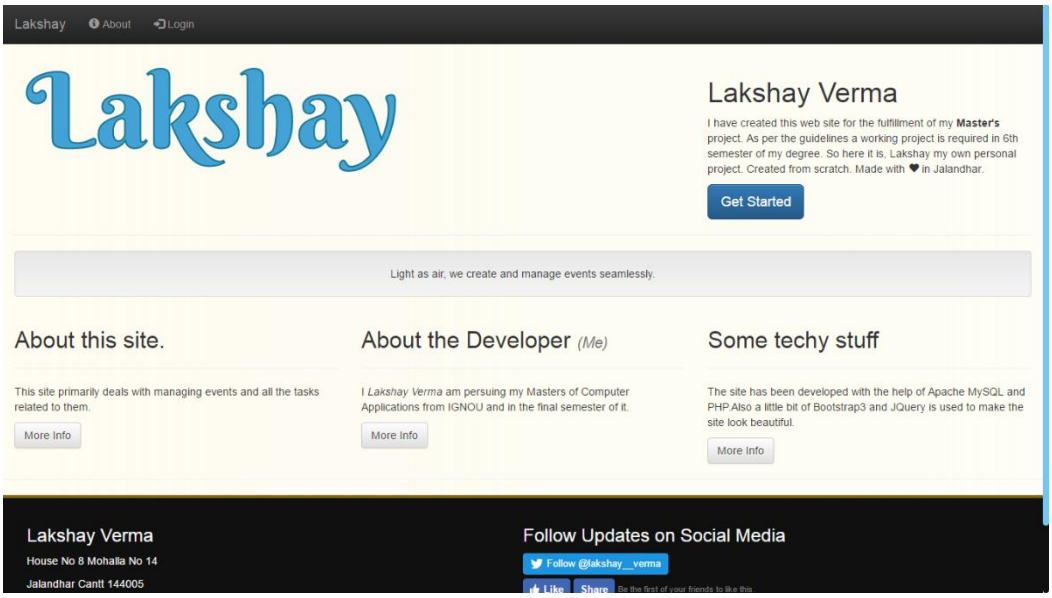


Figure 43 Index.php

The webpage where user will land the first time they visit our website.

Login.php

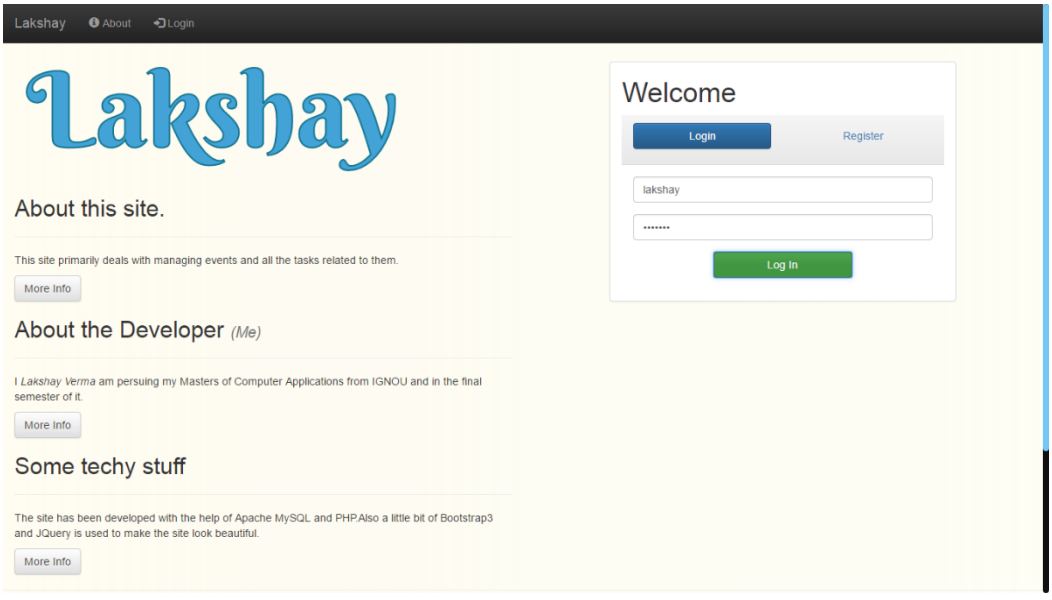


Figure 44 Login.php

To gain access of the portal.

Registeration

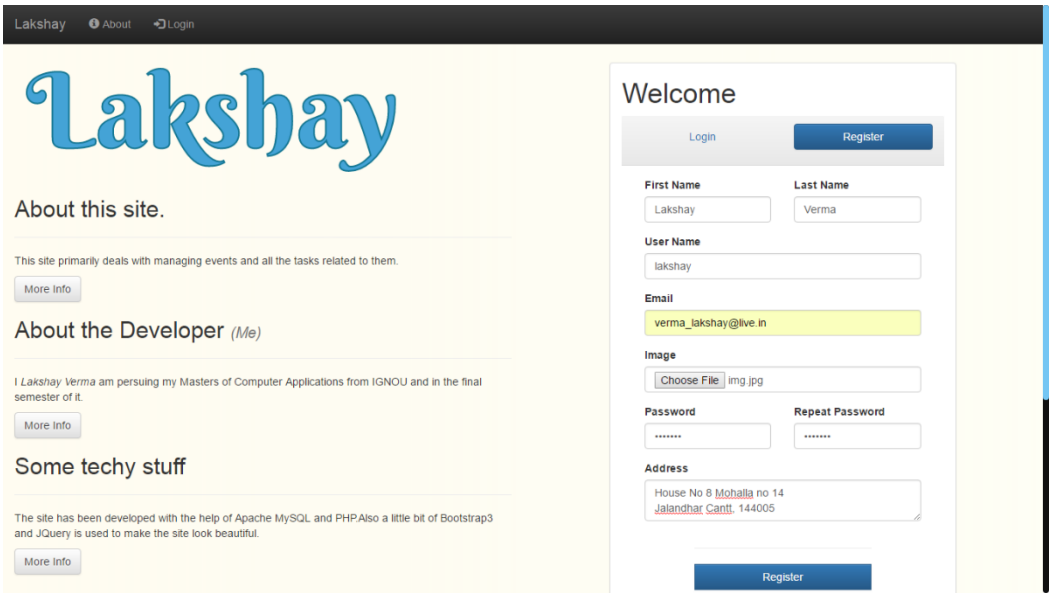


Figure 45 User Registeration

Register as a user to gain access to the portal.

Create New Event

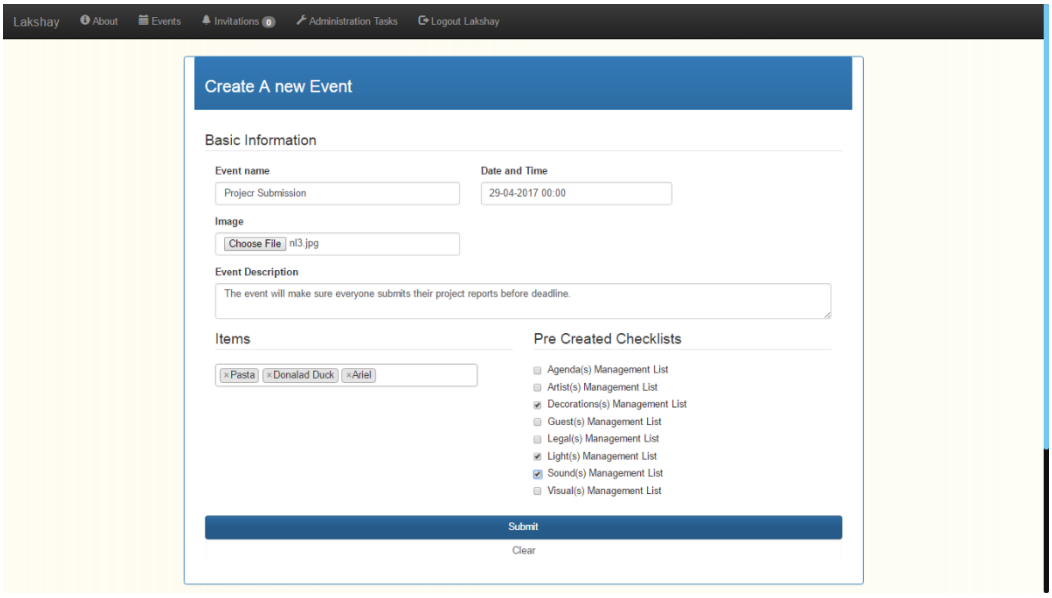


Figure 46 Create a new event

Begin by creating a new event.

Event Details

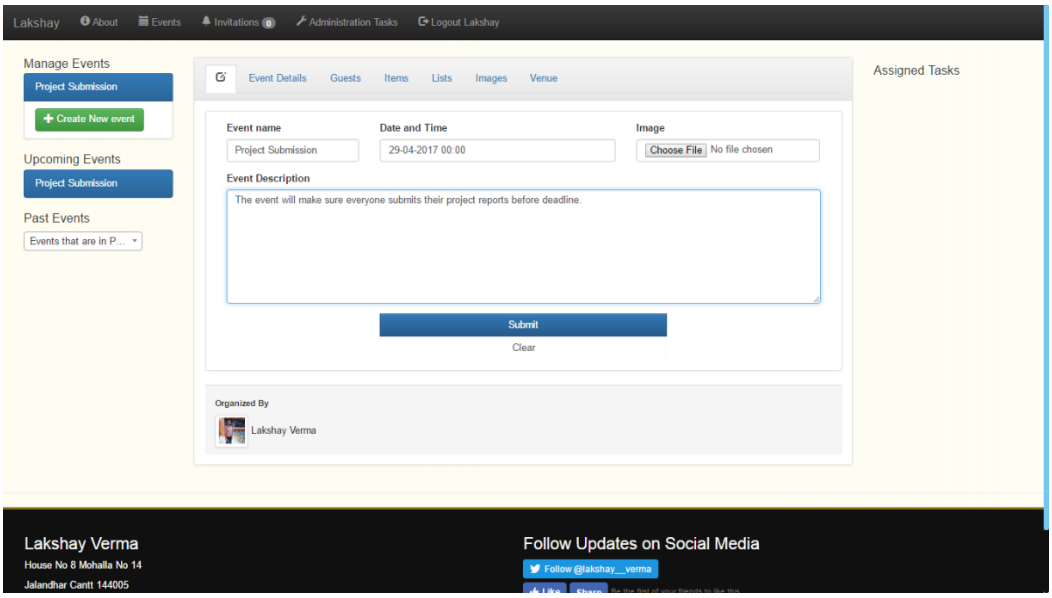


Figure 47 Event Details

View the details related to an event and traverse through other events as well.

Guests List

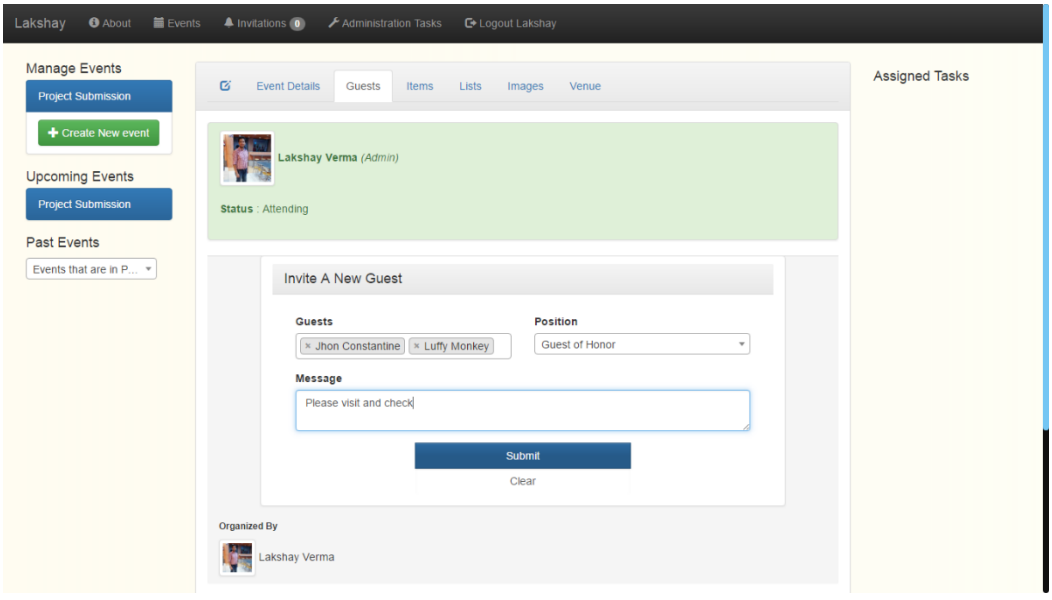


Figure 48 Guest List and Invitation

Guests can be added to the list from this section.

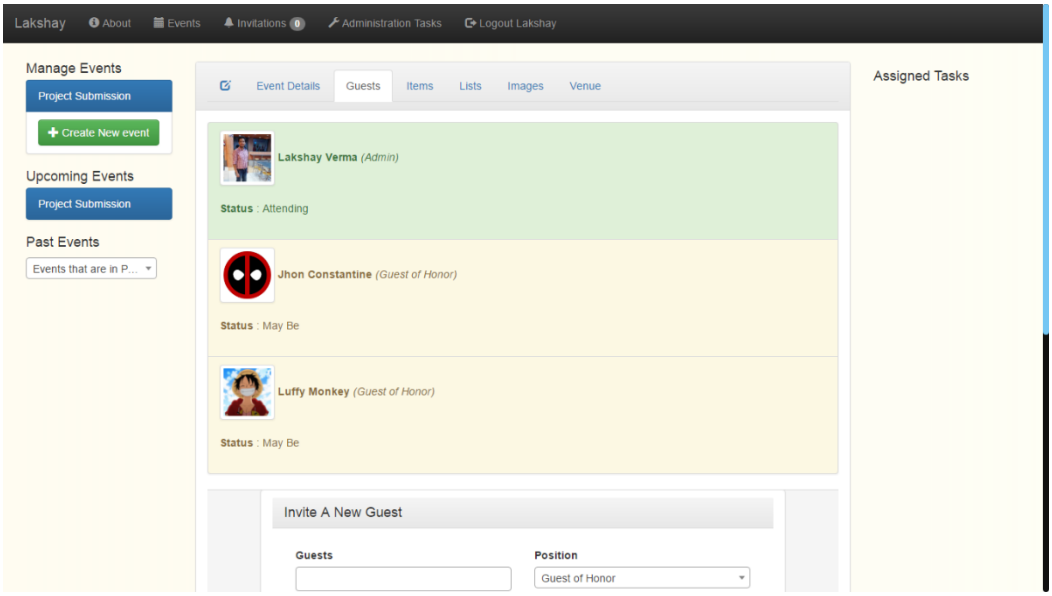


Figure 49 Invited Guests

A list of all the guests and their status in an event.

Event Items

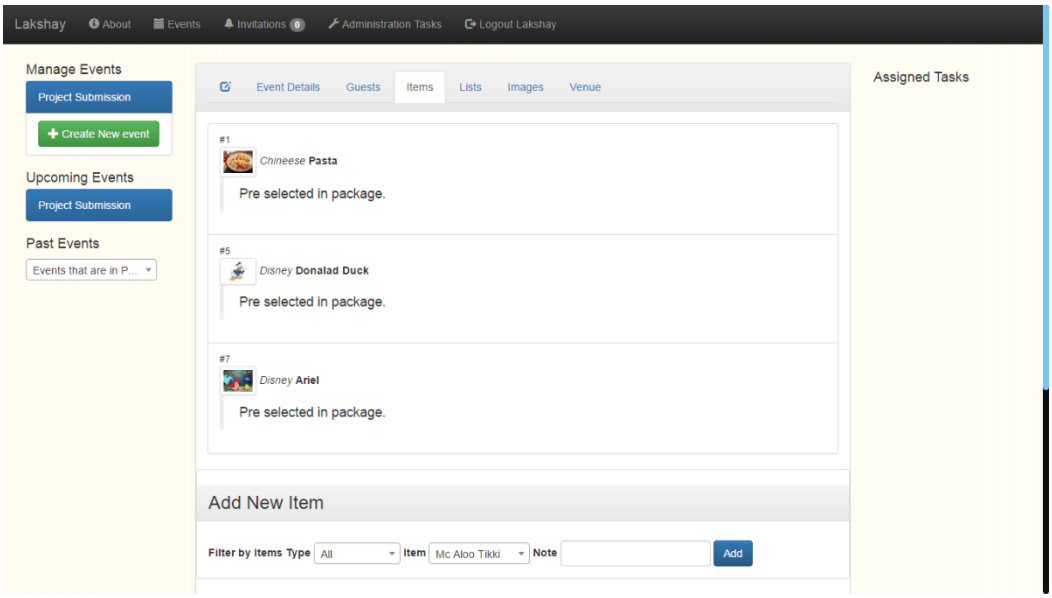


Figure 50 Items in an Event

Items that have been ordered for the event.

Event Checklists

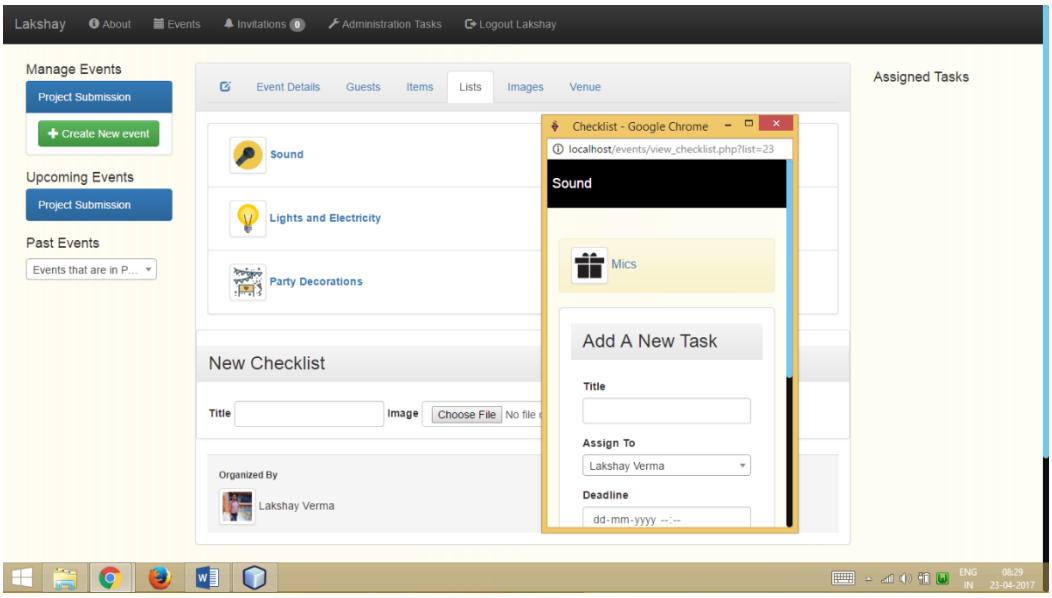


Figure 51 Checklists in an Event and its Tasks

Checklists for event tasks.

Event Schedule

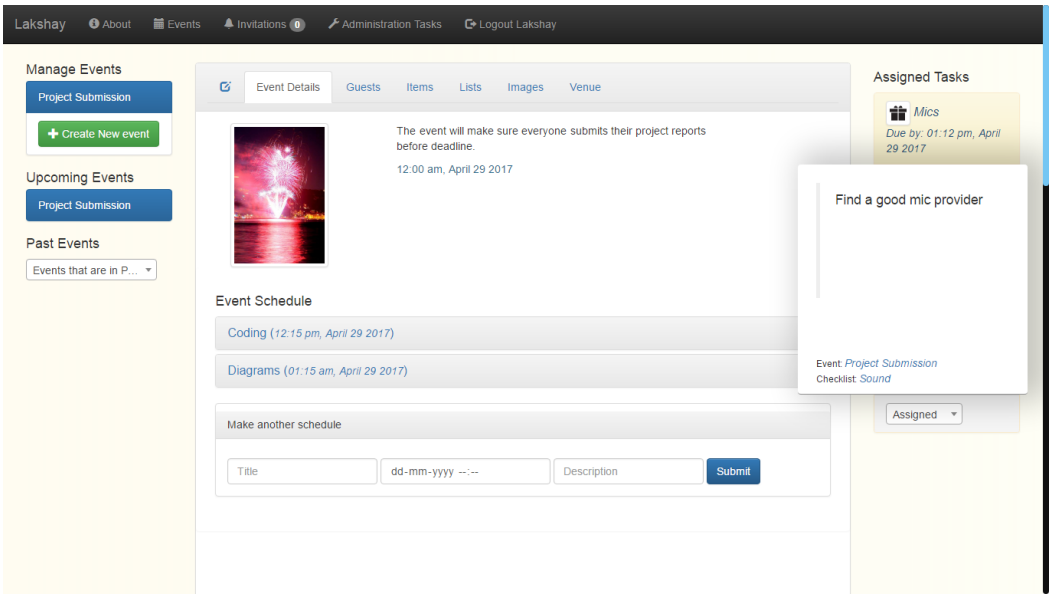


Figure 52 Event Schedule

Schedule of the current event with its details.

Event Venue

Venue Selection

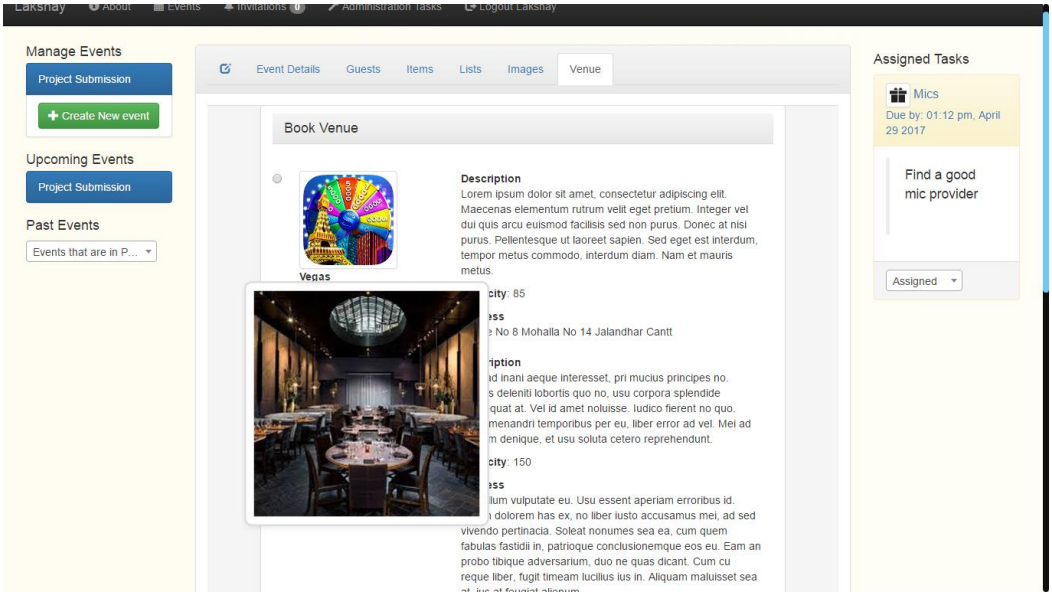


Figure 53 Venue Selection

Selecting a venue for an event where everyone will be gathered.

Venue Details as Displayed to Others

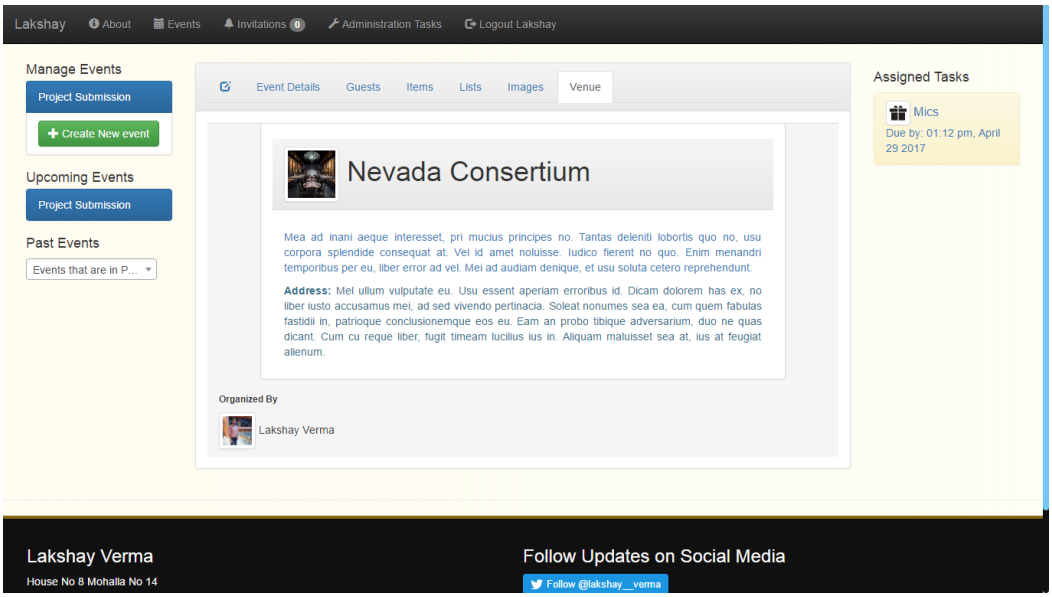


Figure 54 Event Venue display to other Guests

Guests can see this to know the address and other details regarding the venue of the event.

Event Review

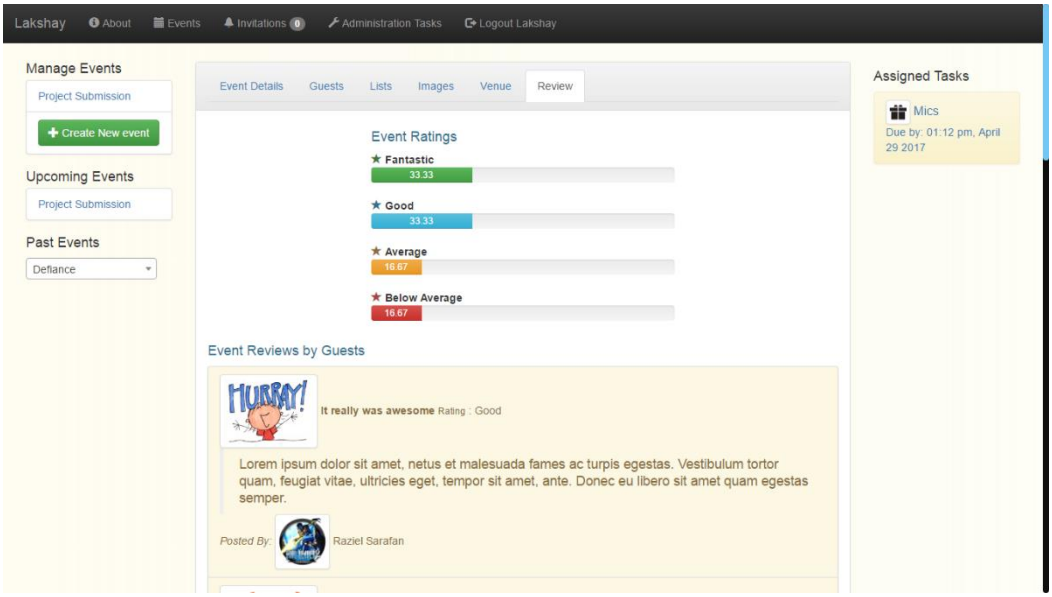


Figure 55 Event Review

After an event ends all the guests can rate the event for report generation and credibility purposes. Also this ensures the goodwill of an event manager.

Event Item Review

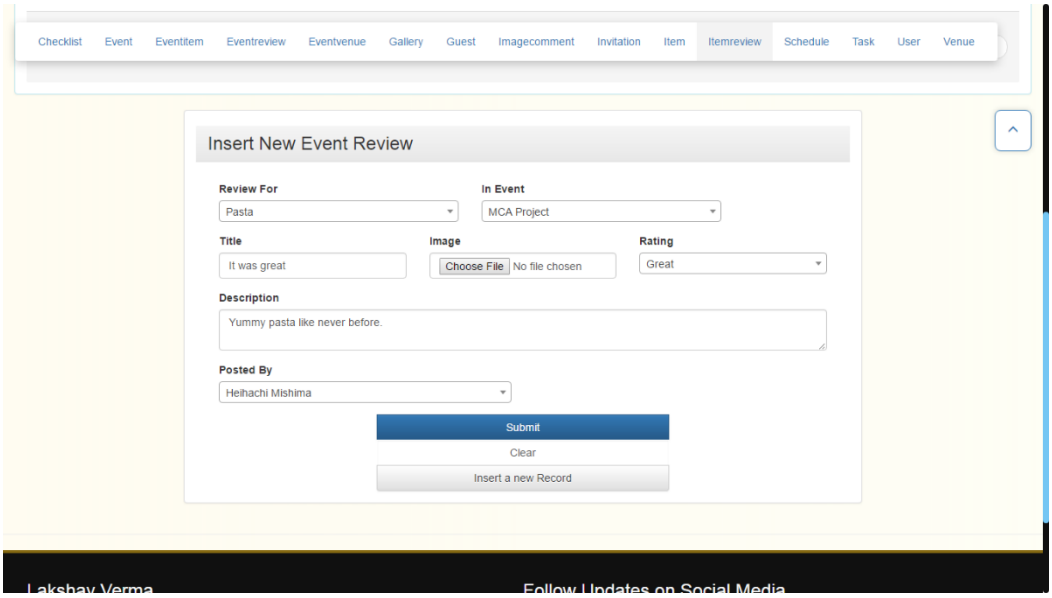


Figure 56 Item Review

Event managers have the unique facility to rate individual items as well so other can know an item is a good choice for their event.

Event's Image Gallery

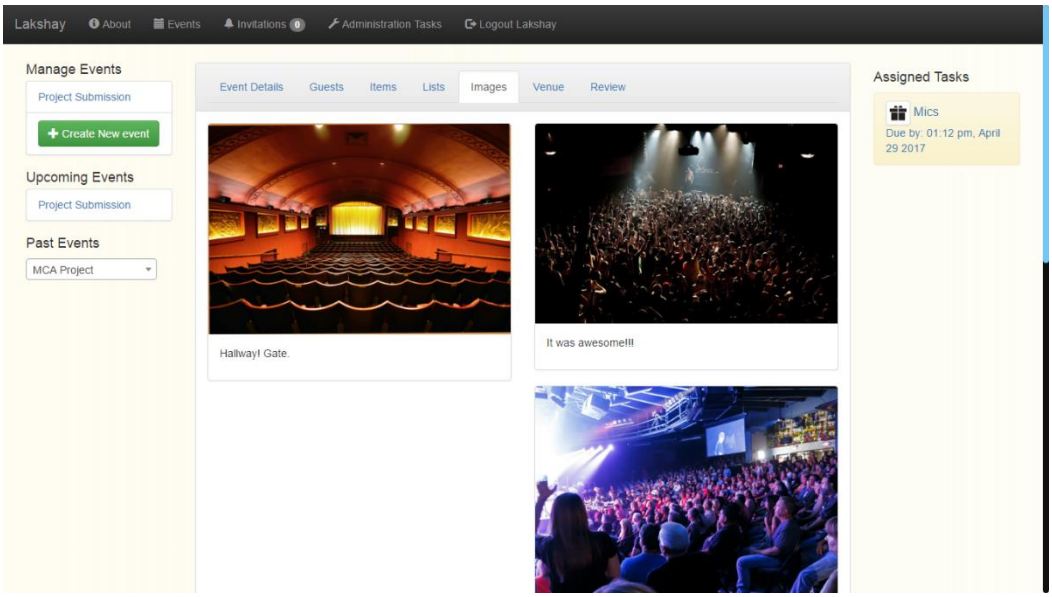


Figure 57 Event gallery

Something extra to showcase the event.

Event Image Viewer

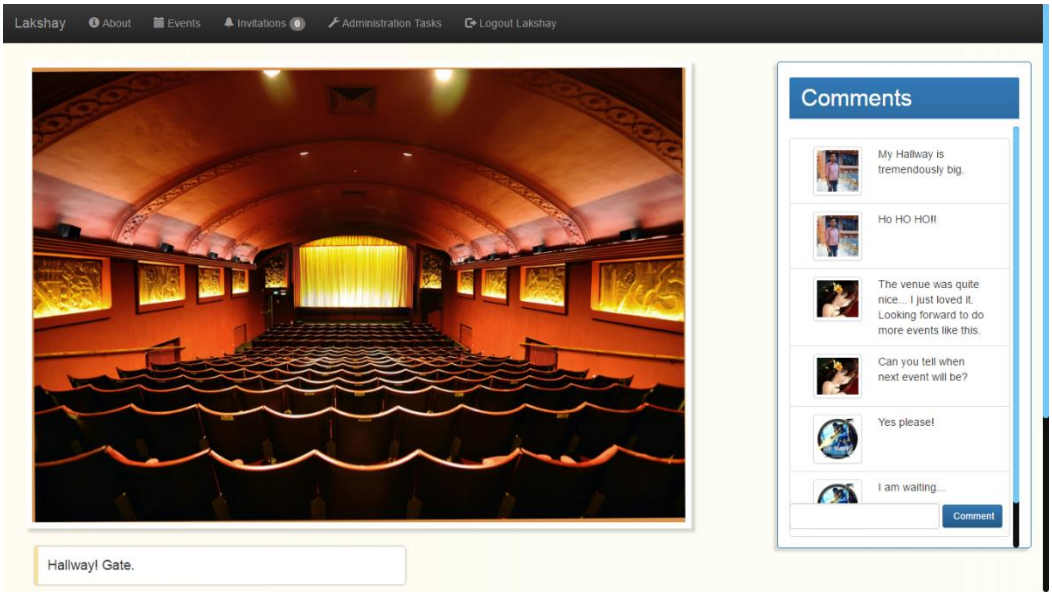


Figure 58 Image viewer

To view all the images separately and comment on them as well.

Cost Estimation

The constructive cost model was developed by Barry W.Boehm in the late 1970s[1] and published in Boehm's 1981 book Software Engineering Economics[2] as a model for estimating effort, cost, and schedule for software projects. It drew on a study of 63 projects at TRW Aerospace where Boehm was Director of Software Research and Technology. The study examined projects ranging in size from 2,000 to 100,000 lines of code, and programming languages ranging from assembly to PL/I. These projects were based on the waterfall model of software development which was the prevalent software development process in 1981.

Basic COCOMO

Basic COCOMO compute software development effort (and cost) as a function of program size. Program size is expressed in estimated thousands of source lines of code (SLOC, KLOC).

COCOMO applies to three classes of software projects:

- **Organic projects** - "small" teams with "good" experience working with "less than rigid" requirements
- **Semi-detached projects** - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements
- **Embedded projects** - developed within a set of "tight" constraints. It is also combination of organic and semi-detached projects.(hardware, software, operational, ...)

Mode	A	B	C	D	KLOC	Effort	Duration	Staffing
Organic	2.4	1.05	2.5	0.38	4.8	12.459	6.5	1.91

Explanation: The coefficients are set according to the project mode selected on the previous page, (as per Boehm,81). The final estimates are determined in the following manner:

effort = a*KLOCb, in person/months, with KLOC = lines of code, (in the thousands), and:

duration = c*effortd, finally:

staffing = effort/duration

Testing

Test reports for basic test cases

Test Scenario	Test Description	Input	Expected Outcome	Outcome
User Login	Upon Input of Correct Username Password combination the user should be allowed to enter the system	Username Password	User Page for Logged in account	PASSED
User Login	Upon Input of Correct Username Password combination the user should be allowed to enter the system	Username Password Incorrect	Incorrect Credentials Messsage	PASSED
User Registeration	First time users fill out a simple form to gain access into the system	Personal Details Username Password	User Page for newly created Account	PASSED
Event Creation	Fill out Event details and start organising event	Event Title, Date time, and other details	Event Page for newly created event	PASSED
Guest Invitation	Select Users from within the list and send invitations to them	User List	Invitation in their account and their names in guest list	PASSED
Checklist Creation	Create checklist for an event	Title and Event the checklist is for	Checklist item displayed in the event	PASSED
Task Creation	Add a task into an checklist and assign the task to other members	Task title, description, assigned to, deadline	Task assignment to correct member and displayed in their portal	PASSED
Image Upload	Upload an Image to event's gallery	Image	Image displayed in event's gallery	PASSED
Image Comment	Comment on an image	Comment text	Comment shown in the	PASSED

			comments panel of Image viewer under User's name who posted the comment	
Logout	Exit the portal by ending session	Null	Secure pages require login again	PASSED
Add Schedule	Select an event and add a schedule to it	New schedule's time	Schedule added in the event.	PASSED

System Security measures

Database / data security

Database can be secured in multiple ways and the most basic are.

- Data types for every column so wrong or error prone data never gets entered
- Using Database users
- Applying Constraints
- Use of Relation between two tables with foreign key concept rather than duplicating data to make database vulnerable towards data attacks.

It is very important for any system that the system is secure from threats like Hackers and Unauthorized edits. For that purpose as discussed earlier in this report we have provided the system with two types of users.

Creation of user profiles and access rights

1. Admin
2. User

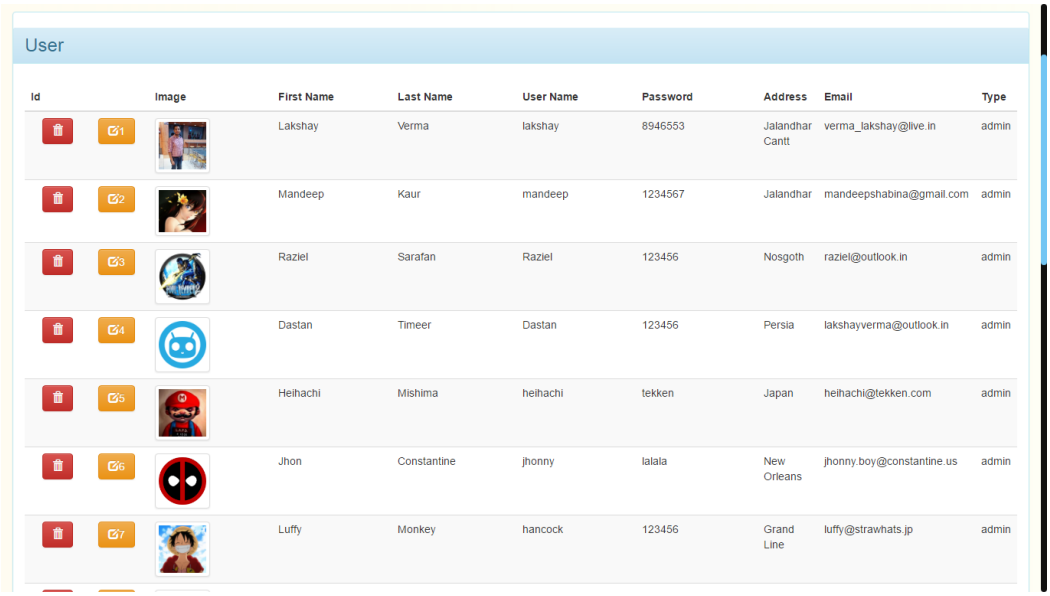
User

This user has the rights to create update and view events, their schedules and other details without any issues. Also they can update their profiles.

Admin

An admin has all the access of a User plus the features like deletion of rows from the database. Insertion of new venues and generating reports as well.

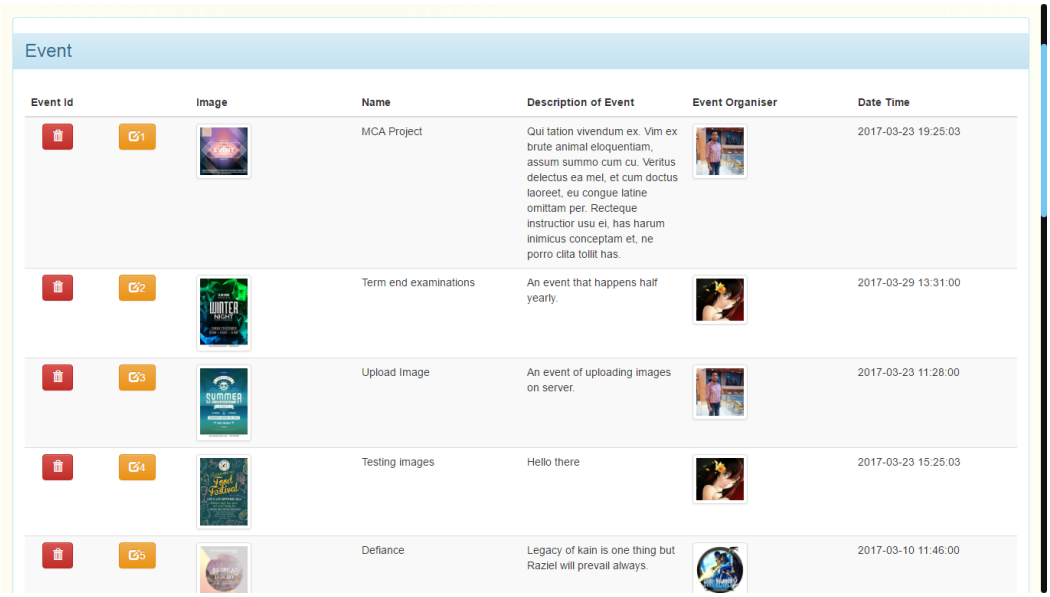
Reports



User

Id	Image	First Name	Last Name	User Name	Password	Address	Email	Type
1		Lakshay	Verma	lakshay	8946553	Jalandhar Cantt	verma_lakshay@live.in	admin
2		Mandeep	Kaur	mandeep	1234567	Jalandhar	mandeepshabina@gmail.com	admin
3		Raziel	Sarafan	Raziel	123456	Nosgoth	raziel@outlook.in	admin
4		Dastan	Timeer	Dastan	123456	Persia	lakshayverma@outlook.in	admin
5		Heihachi	Mishima	heihachi	tekken	Japan	heihachi@tekken.com	admin
6		Jhon	Constantine	jhonny	lalala	New Orleans	jhonny.boy@constantine.us	admin
7		Luffy	Monkey	hancock	123456	Grand Line	luffy@strawhats.jp	admin

Figure 59 Users



Event

Event Id	Image	Name	Description of Event	Event Organiser	Date Time
1		MCA Project	Qui tation vivendum ex. Vim ex brute animal eloquentiam, assum summo cum cu. Veritus delectus ea mel, et cum doctus laoreet, eu congue latine omittam per. Recteque instructior usu ei, has harum inimicus conceptam et, ne porro clita tollit has.		2017-03-23 19:25:03
2		Term end examinations	An event that happens half yearly.		2017-03-29 13:31:00
3		Upload Image	An event of uploading images on server.		2017-03-23 11:28:00
4		Testing images	Hello there		2017-03-23 15:25:03
5		Defiance	Legacy of kain is one thing but Raziel will prevail always.		2017-03-10 11:46:00

Figure 60 Events

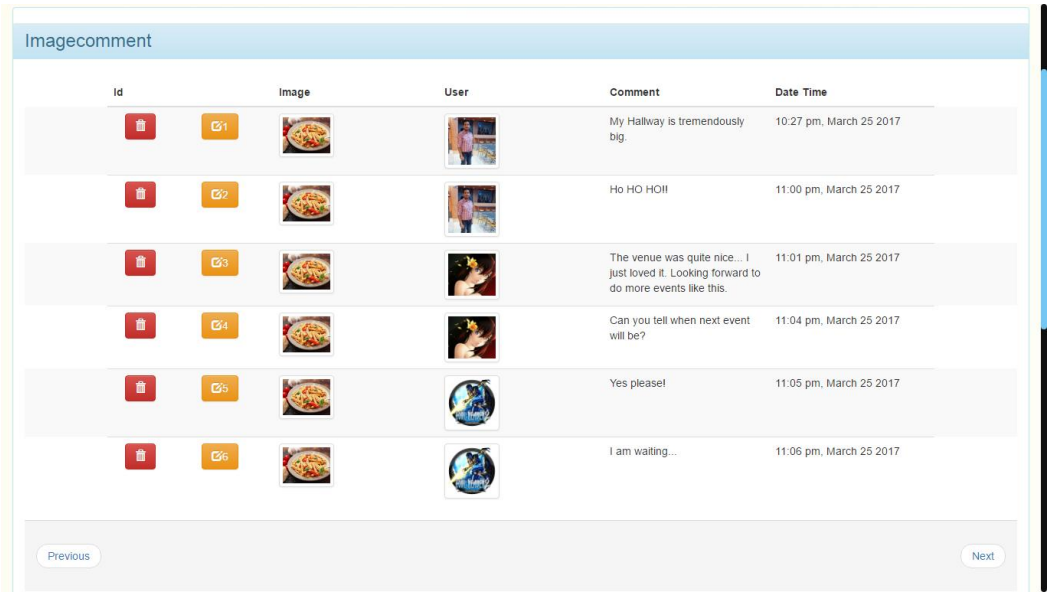


Figure 61 Image Comments

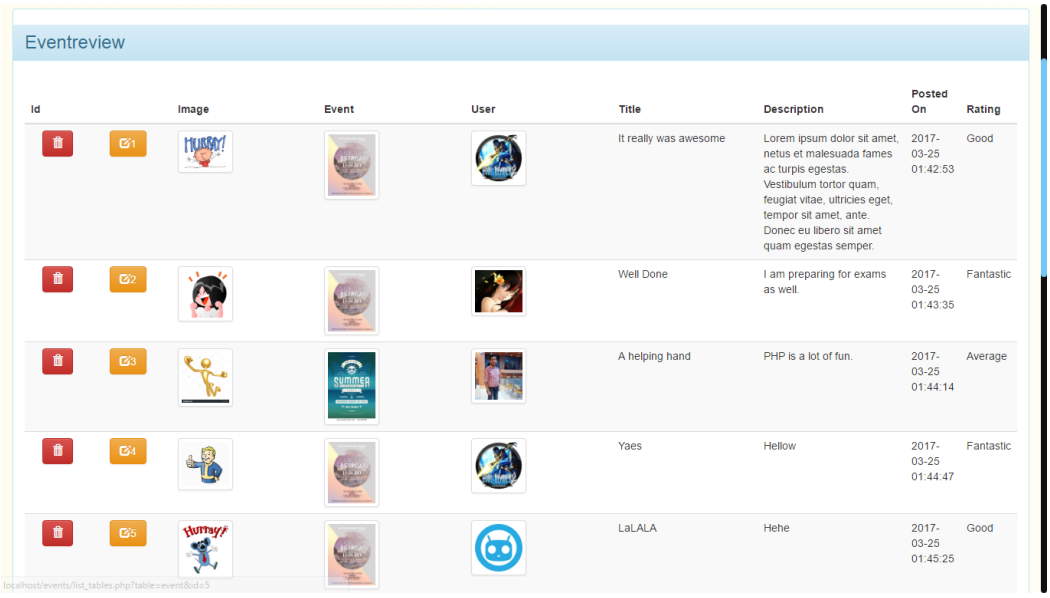


Figure 62 Event Review

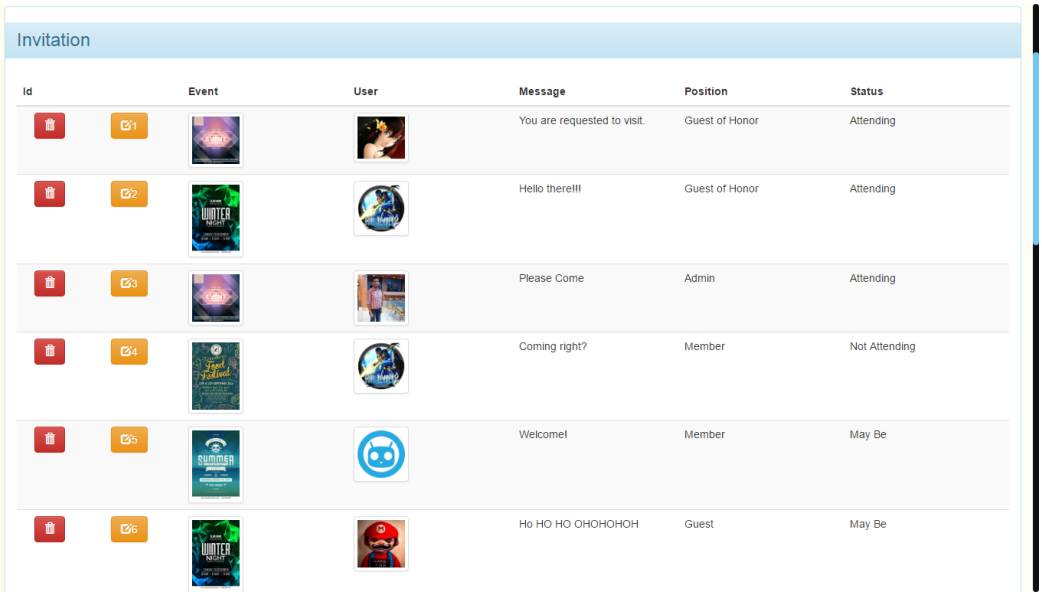


Figure 63 Invitations

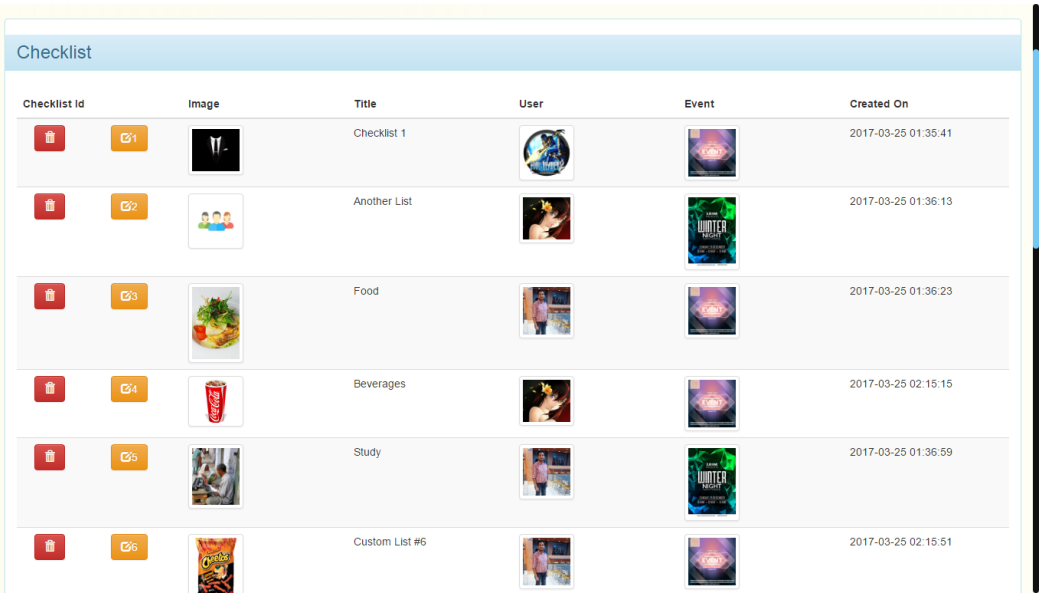


Figure 64 Checklist

Future Scope and further enhancement of the Project

The project is specifically meant to aid in the process of event management and we already have tried to implement the most features in the given time. But there always is room for improvement and the following features can be added in the future.

- Item booking to outside vendors
- Venue Google Map location
- Social media notifications to guest
  - Google Plus
  - Facebook
  - Twitter
- Email sending
- Signup using social media account
- Captcha for improved security

Also the site can be made into an Android application to make the process a lot more easy and on the go.

Bibliography

1. HTML5 (Hyper Text Markup Language version 5): http://w3c.org/html.
2. CSS (Cascading Style Sheet version 3): http://w3c.org/css,
3. PHP: http://php.net.
4. StackOverFlow (Coding Community): http://stackoverflow.com.
5. The New Boston (Tutorials for Technologies Used).
6. Wikipedia(The free encylopedia): http://en.wikipedia.org.
7. Software Engineering a Practitioner Approach 5th Edition by Roger S. Pressman, Ph.D.
8. Google (Search Engine for various Queries): http://google.co.in.
9. Tool for estimating Cost : http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/kutcher/kutcher.html