**Meeting Scheduler Tool**

**Team Members List:**

1. Venkhata Prabha Varadharajan-1001101509
2. Nispand Mehta-1001163146
3. Pooja Pandey-1001153717
4. Arun Gopinathan-1001160219

**Section 1 – Introduction**

The project aims at creating a Meeting Scheduler android application. A meeting can be scheduled for which an invite can be made with the date and time of the meeting for the potential participants of the meeting. The location of the meeting is decided based on all the users’ current location and calculating a common meeting point. Based on the availability of the participants the organizer chooses the location and the date and time of the meeting. A potential user is required to register with the APP server. After registration the user can login with their credentials to create an invitation for the meeting. The invite can be sent to the list of potential participants who can choose to be a part of the meeting based on their schedule. The organizer is notified if the participants have accepted or declined the invitation. The participants will also be able to view the list of all meeting invites.

This report contains a brief explanation of the applications that exist currently. It also explains the application’s component design and how the application is presumed to function. The communication overhead for the tool is also explained.

**Section 2 – Discussion on related real systems**

1. **Google Calendar**



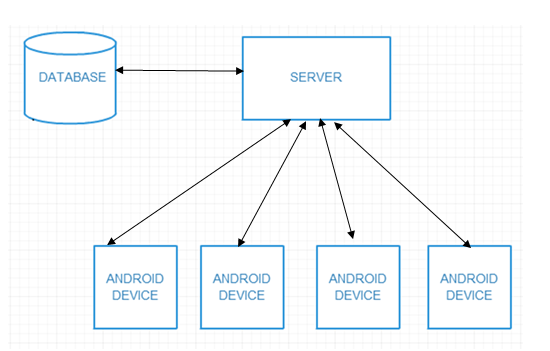
Google Calendar is an example of a meeting scheduler. It helps to understand who the organizer of the meeting is, the participants of the meeting, gives details on where the meeting is to be held and also states the time at which the meeting will be held. It offers participants choices (YES, NO and MAYBE) to attend the meeting. It also shows whether the participants will attend the meeting.

1. **Meeter Easy Meeting**



This is an iOS application named Meeter Easy Meeting which lets the organizer pick up the time slot of the meeting and enables the organizer to know whether the participants are available at the given time slot by checking their calendars.

**Section 3 – System Overview**



**i. Android Device**

The android users who have installed the Meeting Scheduler application can schedule meetings with other android users who have installed Meeting Scheduler application. An invite is created with the details of the participants, location, date and time at which the meeting is likely to be scheduled at. The participants who are invited can choose to either accept or reject the scheduled meeting.

**ii. Server**

Our application is running on Apache Tomcat server. On completion of registration, the details are stored in the MySQL database. A registered user can then send invites to potential participants using the participants’ registered email addresses.

**iii. Interactions**

The android device interacts with the web service which is deployed in the server. The web service is responsible for storing, retrieving and processing the operations of the meeting scheduler. The client interacts with the server using the web service.

**Section 4: Detailed Design**

**Components:**

* 1. **Client(Application)**
  2. **Application Server**
  3. **Database Server**

1. **Client(Application Part)**

**Android Device** – The meeting scheduler application is installed in the android device. The application serves as a client. The User Interface contains registration of the user, account credentials to login, schedule a meeting, accept or decline a meeting. The client will communicate with the server to perform all the operations.

Application contains the following features:

* Login
* Propose Meeting
* Schedule Meetings
* Accept/Reject Meetings
* Register for Account
* Status of Participants (Accepted/Rejected)

|  |  |
| --- | --- |
| **UC1 : Propose Meeting** | |
| Actor : User | System : Meeting Scheduler |
|  | (0) System Displays the Login Page |
| (1)TUCBW user logs in using his/her registered email ID and password | (2) System authenticates the user and directs the user to the main menu page |
| (3)On the main menu page the user clicks on the propose meeting menu. | (4)System prompts the user to enter all the details like Agenda, Date, Location and proposed timings for the meeting |
| (4)The user will click on next button | (5)The system displays the page wherein the user can add the email ids of the probable participants for the meeting |
| (6) The user clicks on add invites and enters the email addresses of the participants and clicks on create meeting button. | (7) The system will store the data in the database and will send the notification to all the invitees for the meeting |
| (7) TUCEW the user showing the notifications meeting registered successful. |  |

|  |  |
| --- | --- |
| **UC2:Accept/Reject Meeting** | |
| **Actor : User** | **System : Meeting Scheduler** |
|  | (0) System displays the Login Page |
| (1)TUCBW user logs in using the registered email id and password | (2) System authenticates the user and directs the user to the main menu page |
| (3)User clicks on notifications button to check the list of meetings | (4) System will show the user all the proposed meeting details like agenda, date, and location and proposed timings for the meeting. |
| (5)User can mark his/her availability for the proposed timings and can provide the location preferences if he accepts/rejects the meeting invite. | (6)System will record Users timing and location preferences and stores the values in the database. |
| (7)TUCEW user shown confirmation message successfully. |  |

|  |  |
| --- | --- |
| **UC1:Login** | |
| **Actor : User** | **System :Meeting Scheduler** |
|  | (0) System displays the Login Page |
| (1) User enters the username and password | (2)System validates the user and if valid directs it to the next page. |
| (3)TUCEW the confirmation message login successful if the user is registered else login failed message will be displayed. |  |

|  |  |
| --- | --- |
| **UC3: Register User** | |
| **Actor : User** | **System: Meeting Scheduler** |
|  | **(0)**System displays the Login Page and shows register option. |
| **(1)**TUCBW user clicking on Register Link | **(2)**System redirects the user to Registration page and prompts the user to fill the details. |
| **(3)**User adds the details like email id, first name, last name and password and will click on the submit button. | **(4)**System will send the user details to the application server in order to store them in the database. |
| **(5)**User will mark his/her availability for the proposed timings. | **(6)**Application will return success message if the user is successfully registered, in case the user is already registered it will return a failed message. |
| **(7)**TUCEW user shown confirmation message successfully. |  |

**ii) Application Server**

The Application server which is Apache Tomcat hosts a Java web service. This web service will implement all the business logic that is required for the android application to register a user, login, schedule a meeting, and accept a meeting invite or reject a meeting invite. The Web Service will be responsible for all the operations and the all the data related to these operations will be stored in the database by the Web Service.

**iii. Database**

We have chosen MYSQL as our database wherein all the data regarding the App will be stored.

The Database will contain following tables which will be shared between all the clients.

1. **Login table**

|  |  |
| --- | --- |
| **Login table** | |
| UserId | Int(11) Primary Key |
| FirstName | Varchar(50) |
| LastName | Varchar(50) |
| MavEmail | Varchar(50) |
| Password | Varchar(50) |

1. **Proposed Meeting**

|  |  |
| --- | --- |
| Proposed Meeting | |
| MeetingId | Int(11) Primary Key |
| UserId | Int(11) Foreign Key |
| Agenda | Varchar(100) |
| MeetingName | Varchar(20) |
| Location | Varchar(20) |

1. **Participants**

|  |  |
| --- | --- |
| Participants | |
| ParticipantId | Int(11) Primary Key |
| MeetingId | Int(11) Foreign Key |
| Accepted | Varchar(5) |
| UserEmailId | Varchar(50) |

|  |  |
| --- | --- |
| **MeetingDates** | |
| MeetingDateId | Int(11) Primary Key |
| MeetingId | Int(11) Foreign Key |
| MeetingDate | date |
| MeetingStartTime | time |
| MeetingEndTime | time |

|  |  |
| --- | --- |
| **Response** | |
| ResponseId | Int(11) Primary Key |
| ParticipantId | Int(11) |
| MeetingId | Int(11) |
| MeetingDateId | Int(11) |
| Presponse | Varchar(10) |

**iv. Challenges:**

**Interacting with Google Maps API -** Interacting with the Google Maps API to get thesuggestion of list of places nearby the user as location for the meeting.

**Section 5: Implementation**

**i. Tools**

|  |  |
| --- | --- |
| Android Studio | The IDE for development of Android |
|  | Application. |
| Jersey | For the development of rest Webservice |
| Apache Tomcat | For hosting the web application |
|  |  |
|  |  |
| simple framework | Framework for XML Serialization. |

ii. **Work Split:**

|  |  |  |
| --- | --- | --- |
| Web Services Implementation | | Arun, Prabha |
| Android | Application | Nispand , Pooja |
| Implementation |  |  |
| Application Integration |  | Arun, Nispand |

**Section 6: Theoretical / Simulation Study**

**i. Communication Overhead**

The communication overhead that we faced while developing the App is to get the acceptance and rejection notification from all the probable participants and their location preferences in case they accept the meeting invite.

1. **Storage Overhead**

Storing the accept and reject meeting invites notifications from the probable participants and their location preferences in case of acceptance in order to calculate the common venue for the meeting was the storage overhead that we faced.

**Section 7 – Future Enhancements**

In the application implementation the future goals are to provide estimates on whether the location of the meeting is feasible or not based on the availability of the location also we can calculate the suitable meeting time based on the user preferences automatically instead of doing it manually.