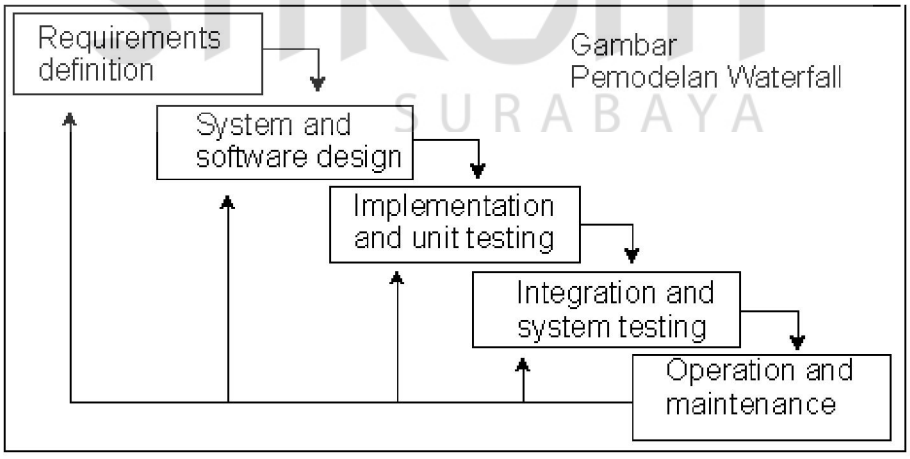
BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis sistem

Untuk membangun aplikasi ini, dilakukanlah analisis proses kerja rapat yang ada pada saat ini untuk mengetahui kendala-kendala pada proses tersebut. Selanjutnya dari analisis kendala-kendala tersebut digunakan untuk membangun aplikasi pendukung. Pada pengembangan aplikasi terdapat banyak model pengembangan sistem dan aplikasi salah satunya adalah model *waterfall*, yang akan digunakan dalam pengembangan aplikasi ini.

Dalam model pengembangan perangkat lunak dengan *waterfall*, terdapat beberapa langkah yang dapat dilakukan antara lain: (i) analisis kebutuhan, (ii) desain *software* dan sistem, (iii) Implementasi dan pengujian sistem, (iv) integrasi sistem, dan (v) *maintenance* sistem (Pressman, 2002). Model pengembangan *waterfall* dapat digambarkan dengan diagram pada Gambar 3.1 berikut ini.



Gambar 3.1 Waterfall Model

3.1.1 Identifkasi Masalah

Untuk melakukan identifikasi masalah maka dilakukan observsasi pada PT. Garasilabs Manivesta Surabaya. Observasi ini dilakukan untuk menggali informasi dan menganalisa proses bisnis rapat yang terintegrasi dengan model pengembangan perangkat lunak pada PT. Garasilabs Manivesta, *scrum methodology*. Pada tahapan ini, informasi yang dikumpulkan adalah proses finalisasi *sprint* dari sebuah proyek perangkat lunak, penentuan *backlog* / *release backlog* dari sebuah perangkat lunak, penentuan daftar fitur pada sebuah *sprint*, serta perkembangan pengerjaan sebuah tugas-tugas pada setiap harinya.

Dalam melakukan pengembangan produknya, PT. Garasilabs Manivesta menggunakan salah satu metode dari model pengembangan *Agile*, *scrum*. Metode ini mengharuskan tim untuk melakukan konsolidasi dalam bentuk rapat yang disebut *Scrum Meeting*. *Scrum Meeting* dilakukan dengan batas waktu yang telah ditentukan. Pada saat melakukan *scrum meeting*, para staf yang terlibat memasuki ruangan *conference chat* yang telah dibuat oleh moderator. Di dalam ruangan *conference* itulah, kegiatan rapat berlangsung. Topik rapat akan diberikan oleh moderator kepada para peserta melalui interaksi *chat*. Peserta rapat yang terputus dari ruangan ataupun baru bergabung di dalam rapat, tidak dapat mengikuti topik yang tengah dibicarakan.

Peserta rapat kemudian mengajukan gagasan yang berhubungan dengan topik yang telah ditetapkan oleh moderator rapat, beserta penjelasan-penjelasan yang mendukung. Peserta yang ingin melakukan interupsi ataupun ingin menambahkan penjelasan yang mendukung gagasan peserta lainnya, juga menuliskannya pada kotak *chat* di dalam ruangan yang sama dari media

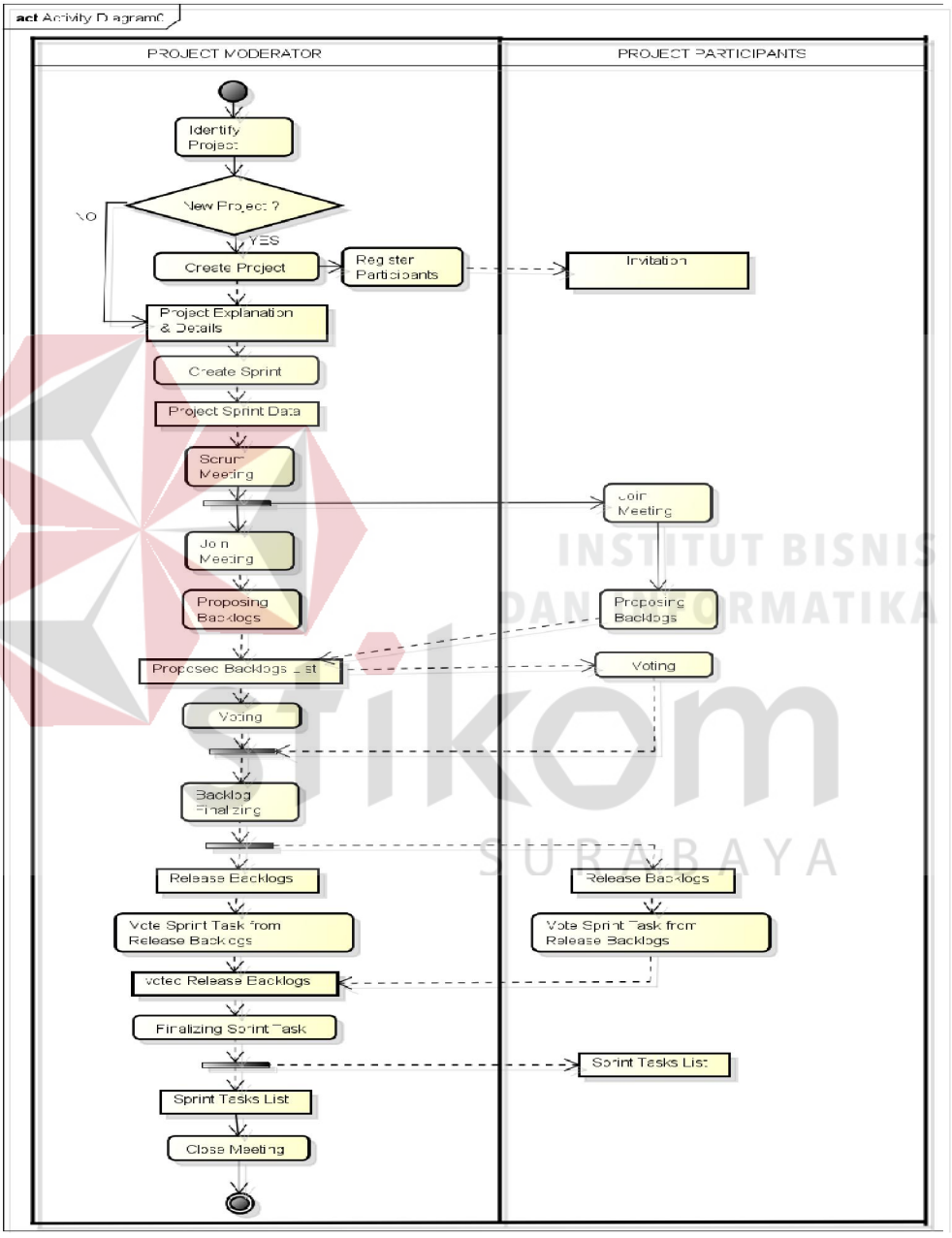
conference chat ini. Kegiatan vote juga dilakukan di dalam media ini, dengan menggunakan fasilitas *chat* yang sama. Vote dilakukan dengan cara menelusuri kembali log *chat* yang telah terjadi selama berlangsungnya rapat lalu dipilah menjadi vote *items* yang akan menjadi objek voting. Kegiatan voting ini kemudian menghasilkan beberapa ide yang diutarakan oleh peserta rapat, dan akan direkap lalu didokumentasikan baik untuk menentukan langkah pengembangan berikutnya, maupun fitur atau produk baru dari PT. Garasilabs Manivesta.

Kelemahan media yang digunakan pada saat ini adalah tidak adanya pembeda antara ide utama yang disampaikan dengan interaksi lainnya yang terjadi di dalam rapat, yang menyebabkan informasi di dalam media tersebut menjadi bercampur dan rancu. Hal ini seringkali mempersulit pihak manajemen untuk mengambil keputusan dan melakukan rekap.

Topik yang diberikan oleh moderator rapat dapat berjumlah lebih dari satu dan secara langsung diberikan melalui media ini di dalam kotak *chat* yang sama. Hal seperti ini seringkali membuat para peserta tidak dapat mengikuti jalannya *chat* selama rapat berlangsung karena log *chat* yang memanjang.

Selain itu, kebebasan untuk berinteraksi di dalam media ini, menyebabkan interupsi yang dilakukan oleh masing-masing staf tidak dapat diatur secara tertib, yang memungkinkan peserta rapat yang sedang mengutarakan pendapatnya terganggu, atau sebaliknya, beberapa peserta tidak dapat mengajukan interupsi. Hal ini dirasa kurang menguntungkan terutama bagi peserta yang sedang mengajukan pendapat.

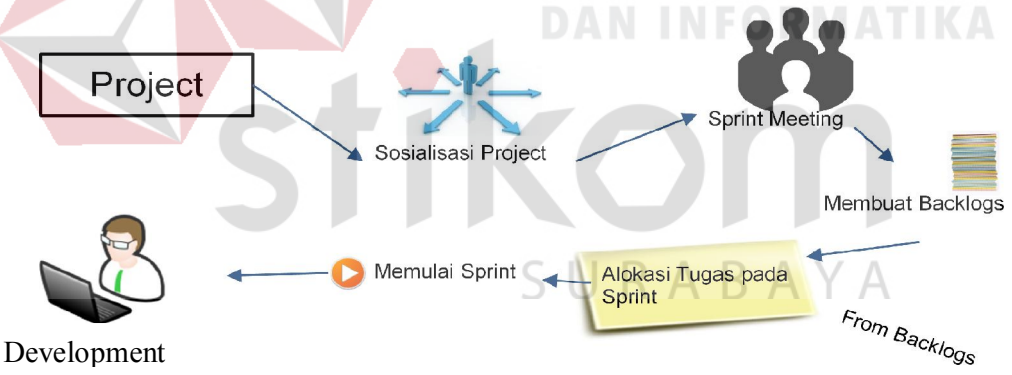
Dari beberapa proses bisnis tersebut dapat digambarkan menjadi *activity diagram* keseluruhan untuk sistem yang ada saat ini seperti pada gambar 3.2 berikut ini.



Gambar 3.2 *Activity Diagram* Proses Rapat pada PT. Garasilabs Manivesta

Dari Gambar 3.2 dapat dilihat proses bisnis yang saat ini terjadi pada PT. Garasilabs. Dalam melakukan pengembangan proyek, diharuskan melakukan kolaborasi pada setiap harinya dalam bentuk rapat. Rapat akan dilakukan dengan mengacu kepada setiap proyek dan didalam rapat tersebut dan akan dibahas mengenai tahapan yang sekarang sedang dilakukan. Pada saat melakukan *scrum meeting*, peserta rapat akan melakukan beberapa kegiatan seperti, pemilihan *backlog*, *voting*, berkolaborasi dalam pemecahan masalah, serta berbagi informasi. Setelah selesai melakukan rapat, maka akan dilakukan finalisasi rapat untuk mencatat apa yang telah dikerjakan dan kesulitan apa saja yang dihadapi dan telah diselesaikan.

Secara umum, *scrum meeting* akan dibedakan menjadi 2 yaitu, *sprint planning* dan *daily scrum* (Dennis, 2007).

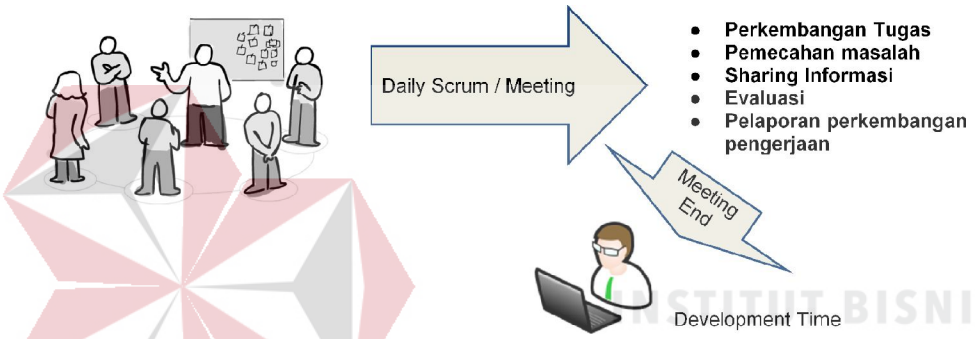


Gambar 3.3 Proses *Sprint Planning*

Gambar 3.3 menggambarkan tentang alur yang ada pada *sprint planning*. Proyek yang telah diterima dari *customer*, akan disosialisasikan terlebih dahulu kepada peserta proyek, baik analis maupun *developer*. Pada saat sosialisasi tengah berlangsung, pimpinan proyek juga harus mengatur *sprint* pertama dari proses pengembangan proyek. Setelah itu, rapat dilakukan untuk membuat daftar

backlogs yang berisi fitur-fitur yang diinginkan maupun menunjang penyelesaian proyek, *customer whislist*, dan beberapa hal lainnya yang berhubungan dengan proyek. Setelah *backlog* telah terkumpul, maka akan dilakukan alokasi tugas terhadap *sprint* yang telah terbuat untuk waktu pengembangan pertama kali dari proyek.

Setelah *sprint* dimuliai, maka setiap harinya akan dilakukan *daily meeting* seperti tergambar pada Gambar 3.4



Gambar 3.4 Proses *Daily Meeting*

Proses rapat harian (*daily meeting*) yang digambarkan pada Gambar 3.4 diatas, menunjukkan beberapa aktivitas yang biasa dan seharusnya dilakukan di dalam rapat harian di dalam tahapan suatu *sprint*. Aktivitas yang dilakukan antara lain pelaporan tugas yang dikerjakan, berbagi informasi, pemecahan masalah bersama, serta perkembangan proses penyelesaian proyek.

3.1.2 Hasil Analisis

Dari hasil analisis permasalahan, didapatkan kelemahan-kelemahan dari sistem yang lama dan untuk mengatasi kelemahan –kelemahan tersebut maka akan dibuat sistem yang dapat menangani permasalahan dan sesuai dengan

kebutuhan pihak PT. Garasilabs Manivesta. Hasil identifikasi masalah pada PT. Garasilabs Manivesta adalah sebagai berikut:

A. Kekurangan sistem yang lama

1. Tidak adanya pembeda antara interaksi percakapan *chat* dengan ide utama yang diajukan pada saat rapat.
2. Topik *sprint* yang diberikan oleh moderator tidak dapat dilihat secara jelas, karena tidak ada letak khusus untuk meletakkannya.
3. Ide dan interaksi yang telah diutarakan tidak dapat secara langsung dirangkum, karena LOG *chat* yang memanjang.
4. Voting dilakukan pada tempat yang sama dengan interaksi percakapan dan ide utama.
5. Interupsi dalam pengajuan ide dan voting tidak dapat diatur.
6. *Review* atas *sprint* beserta tugas-tugasnya masih terbuat secara manual.

B. Kebutuhan pemakai

Kebutuhan dari pengguna yaitu moderator dan peserta proyek pada PT. Garasilabs Manivesta adalah sebagai berikut :

1. Rapat yang dilaksanakan akan berdasar pada setiap *project* yang ditentukan.
2. Jalannya rapat internal dapat diatur oleh moderator dengan menggunakan fungsi moderasi pada saat melakukan pengajuan ide dan *voting*.
3. Ide-ide individu di dalam aplikasi ini akan dibedakan secara *User Interface* dengan interaksi umum di dalam forum diskusi
4. Mempunyai media khusus untuk melakukan *File Sharing*

5. Mencatat riwayat pada setiap transaksi yang telah terjadi seperti percakapan umum, penyampaian pendapat, penyelesaian masalah, dan perkembangan pengerjaan tugas.
6. Dapat mencetak model hasil rapat yang dibutuhkan untuk metode pengembangan *Scrum*

C. Alternatif menggunakan media *chatting* lainnya

Beberapa alternatif media *chatting* yang dapat mungkin digunakan oleh perusahaan sebagai media rapat, seperti *blackberry messenger*, IRC (*Internet Relay Chat*) *client*, dan *Facebook Messenger*. Beberapa media tersebut dinilai belum dapat memfasilitasi kebutuhan pemakai. Beberapa perbandingan yang diambil dari kebutuhan pemakai, digambarkan pada Tabel 3.1

Tabel 3.1 Komparasi alternatif media *chatting* lainnya dengan kebutuhan pengguna

No	Pembanding dari kebutuhan pemakai	Media		
		BBM	*Facebook Messanger*	*IRC client*
1	Rapat yang dijalankan berdasar pada *project.*	Dapat membuat grup dengan topik	Dapat menentukan topik	Tidak dapat menentukan topik
2	Fungsi Moderasi	Tidak ada	Tidak ada	Tidak ada
3	Perbedaan tampilan pada Ide - ide individu peserta rapat	Tidak ada perbedaan	Tidak ada perbedaan	Tidak ada perbedaan
4	Perbedaan tampilan pada pemberian komentar	Tidak ada perbedaan	Tidak ada perbedaan	Tidak ada perbedaan
5	Fungsi *Chatting*	Terdapat Fungsi *chat*	Terdapat Fungsi *chat*	Terdapat Fungsi *chat*
6	*File Sharing*	Dapat mengirim *files*	Dapat mengirim *files*	Dapat mengirim *files*
7	*History* aktivitas	Merekam aktivitas *chatting*	Merekam aktivitas *chatting*	Merekam aktivitas *chatting*
8	*History chats*	Terdapat *history*	Terdapat *history*	Terdapat *history*
9	*History task progresses*	Tidak ada	Tidak ada	Tidak ada
10	*History penyelesaian permasalahan*	Tidak ada fitur penyelesaian masalah	Tidak ada fitur penyelesaian masalah	Tidak ada fitur penyelesaian masalah
11	*Tracking penyelesaian suatu project*	Tidak ada fitur penyelesaian *project*	Tidak ada fitur penyelesaian *project*	Tidak ada fitur penyelesaian *project*

Dari hasil identifikasi yang telah dijabarkan mengenai kekurangan sistem yang ada, alternatif menggunakan media *chatting* lainnya, dan kebutuhan perusahaan, maka dapat disimpulkan bahwa PT. Garasilabs Manivesta membutuhkan sebuah Aplikasi Rapat Online yang merupakan sebuah aplikasi berbasis web yang dapat digunakan sebagai media penunjang kegiatan rapat pada PT.Garasilabs Manivesta. Aplikasi ini dikendalikan oleh seorang staf yang akan dipercaya sebagai moderator rapat oleh perusahaan. Moderator dapat mengatur jalannya rapat internal dengan menetapkan *setting* utama pada ruang rapat sebelum rapat dimulai, serta dapat juga memberikan otorisasi kepada setiap staf peserta rapat di dalam ruangan tersebut untuk melakukan beberapa aksi. Peserta rapat dapat memberikan ide-ide individu di dalam aplikasi ini, selain melakukan interaksi umum di dalam forum diskusi, karena keduanya akan dibedakan secara *User Interface*, sehingga keberadaan interaksi umum dan ide utama dari setiap individu dapat dibedakan dengan jelas. Aplikasi ini akan mencatat riwayat pada setiap transaksi yang telah terjadi seperti percakapan umum, penyampaian pendapat, interupsi, dan *vote process*.

3.2 Melakukan Studi Literatur

Studi literatur tersebut dilakukan untuk mendapatkan landasan teori tentang pengembangan aplikasi rapat online agar pada pelaksanaan pengembangan berdasar pada teori yang semestinya dan dapat mengatasi permasalahan yang dihadapi sebelumnya serta memenuhi harapan dari pengembangan aplikasi. Landasan teori didapatkan dari sumber buku dan jurnal ilmiah mengenai aplikasi rapat online berbasis web. Studi literatur yang dilakukan adalah mendapatkan landasan teori tentang apa saja yang dibutuhkan dalam

pembuatan aplikasi tersebut meliputi rapat, komunikasi, Node.js, *Scrum Development Model*, Socket.io, dan Redis. Berikut beberapa landasan teori yang akan digunakan.

A. Rapat

Menurut Sampebu (2010), rapat merupakan sarana berkumpulnya sekelompok orang untuk menyatukan pikiran, pertimbangan, dan pendapat terhadap suatu urusan, masalah, atau pekerjaan yang diharapkan dapat mencapai suatu mufakat, penyelesaian, dan keputusan. Penyelenggaraan rapat perlu direncanakan dengan baik untuk mencapai tujuan yang diharapkanya itu mufakat, penyelesaian, dan keputusan. Perencanaan rapat meliputi menentukan topik, menentukan tujuan rapat, menentukan peserta atau pimpinan rapat, menyusun agenda rapat, menyiapkan tempat rapat, dan membagikan notulen atau hasil rapat ke setiap peserta.

B. Komunikasi

Komunikasi adalah suatu proses memindahkan informasi dan pengertian (maksud) dari satu orang kepada orang lain (Suprapto, 2006). Komunikasi juga merupakan interaksi antar pribadi yang menggunakan system symbol linguistik, seperti system simbol verbal (kata-kata) maupun nonverbal. Sistem ini dapat mensosialisasikan secara langsung / tatap muka atau melalui media lain seperti tulisan atau visual.

Menurut Rand (2012:12), beberapa alasan pentingnya komunikasi adalah:

1. Komunikasi mendatangkan efektifitas yang lebih besar
2. Komunikasi menempatkan orang-orang pada tempat yang seharusnya.

3. Komunikasi membawa orang-orang untuk terlibat dalam organisasi dan meningkatan motivasi untuk melibatkan kinerja yang baik, dan meningkatkan komitmen terhadap organisasi.
4. Komunikasi menghasilkan hubungan dan pengertian yang lebih baik antara bawahan, kolega, dan orang-orang di dalam organisasi dan di luar organisasi.

Komunikasi menolong orang-orang untuk mengerti perlunya perubahan.Komunikasi meminimalkan permasalahan-permasalahan di dalam keorganisasian seperti konflik, *stress*, demotifasi dan loyalitas.

C. Node JS

Menurut Rauch (2012:26), *Node.js* merupakan suatu teknologi yang menggunakan *javascript*, tetapi bukanlah didesain untuk *browser* pada *client* melainkan dijalankan pada *server*. *Node.js* merupakan sebuah *platform* untuk membangun sebuah aplikasi yang membutuhkan jaringan cepat atau *real time*.

D. Scrum Development Model

Scrum adalah sebuah kerangka kerja sederhana dimana orang-orang yang akan bekerja dapat menyelesaikan masalah-masalah kompleks dan dapat mengembangkan produk dengan nilai setinggi mungkin secara produktif dan kreatif.

Berikut adalah tahapan dari metode *scrum*

1. Product Backlog

   Product Backlog adalah daftar keinginan (*wishlist* / *desirement*) *Product Owner* untuk produk yang akan dikembangkan oleh Tim Pengembang. *Product Backlog* ini disiapkan dan diurutkan oleh *Product Owner* dan harus transparan bagi semua pihak.

2. Sprint

   Durasi dari *Sprint* selalu sama/konsisten sepanjang pengembangan produk berlangsung. Artinya apabila Tim *Scrum* telah sepakat untuk memilih durasi *Sprint* selama 2 minggu, maka sepanjang pengembangan produk panjangnya Sprint selalu konstan 2 minggu. Di akhir *Sprint*, Tim Pengembang harus menyelesaikan sebuah potongan produk (*product increment*) yang dapat digunakan oleh pengguna dan berpotensi untuk dirilis ke lingkungan produksi. Sprint merupakan pembungkus untuk semua event lainnya dalam *Scrum*. *Event-event* lain dalam Scrum semuanya dilakukan didalam *Sprint*. *Event-event Scrum* antara lain adalah:

   a. *Sprint Planning*
   b. *Scrum Meeting*
   c. *Sprint Review*

E. Redis

Redis adalah suatu teknologi penyimpanan data pada *memory* menggunakan metode penyimpanan "*key*" – "*value*" atau lebih sering dikenal dengan nama "*In Memory Database*".

F. Socket.io

Menurut Rai (2013:48), *Socket io* merupakan sebuah modul dari *Node*.js. Dibangun di atas *javascipt* untuk membangun aplikasi *real time*. Mempunyai 2 bagian yaitu modul *client* yang berjalan pada *browser*, dan modul *server* untuk *Node.js*. Pada prinsipnya, *Socket.io* tidak hanya menggunakan protokol *Websocket*, tetapi juga menggunakan JSON dan *Ajax Polling*.

Mekanisme kerja socket io adalah sebagai berikut

1. Client, meminta layanan, langkah :

   Membuka koneksi client ke server, dengan pertama kali membuat socket dengan perintah socket(). Lalu dilanjutkan dengan pengalamatan server, setelah itu komunikasi (mengirim dan menerima data), dapat terjadi dengan menggunakan perintah write() dan read()

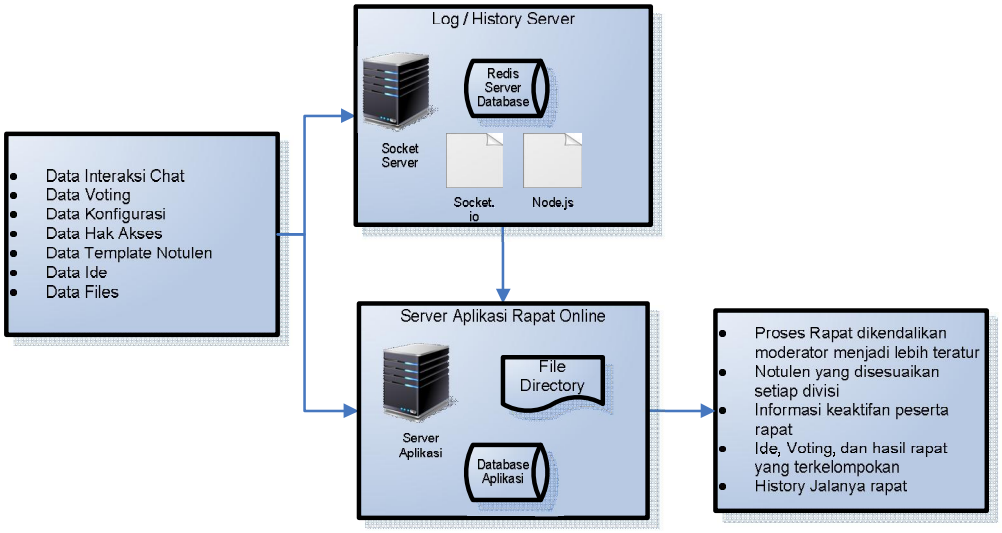
2. Server, menyediakan layanan, langkah:

   a. Melakukan prosedur pembukaan koneksi yang di dalamnya berupa langkah – langkah : membuat socket, mengikat socket, menyiapkan socket menerima koneksi, pengalamatan socket

   b. Looping utama adalah menerima koneksi, dan melakukan komunikasi data (mengirim dan menerima).

3.3 Perancangan Sistem

Tahap perancangan sistem merupakan tahap pengembangan setelah melakukan analisis sistem. Pengguna yang akan menggunakan aplikasi rapat *online* ini adalah Moderator *Project* dan Peserta *Project*. Dalam pengembangannya, aplikasi rapat *online* ini membutuhkan beberapa data yang dapat digambarkan pada Gambar 3.5.



Gambar 3.5 Blok diagram Aplikasi Rapat Online

Dari blok diagram pada Gambar 3.5 telah digambarkan bagaimana proses pencatatan riwayat akan melakukan proses rekap dari berbagai proses yang ada seperti voting, *chating*, serta kolaborasi. Riwayat yang telah direkap akan menjadi suatu hasil dari rapat yang berisi suatu keputusan, dengan hasil rekap riwayat proses-proses jalannya rapat. Sedangkan rincian peran dan tanggung jawab pengguna aplikasi adalah sebagai berikut:

1. **Moderator**

   Moderator adalah pengguna yang bertugas penuh mengawasi, mengendalikan, serta memutuskan topik-topik di dalam suatu rapat. Pengguna ini dapat membatasi penggunaan aplikasi oleh peserta rapat, yang menjadikan keadaan rapat lebih efektif. Moderator juga bertanggung jawab atas pemberian topik dan proses pengambilan suara (voting) di dalam rapat. Moderator juga dapat melakukan hal-hal yang sama dengan peserta.

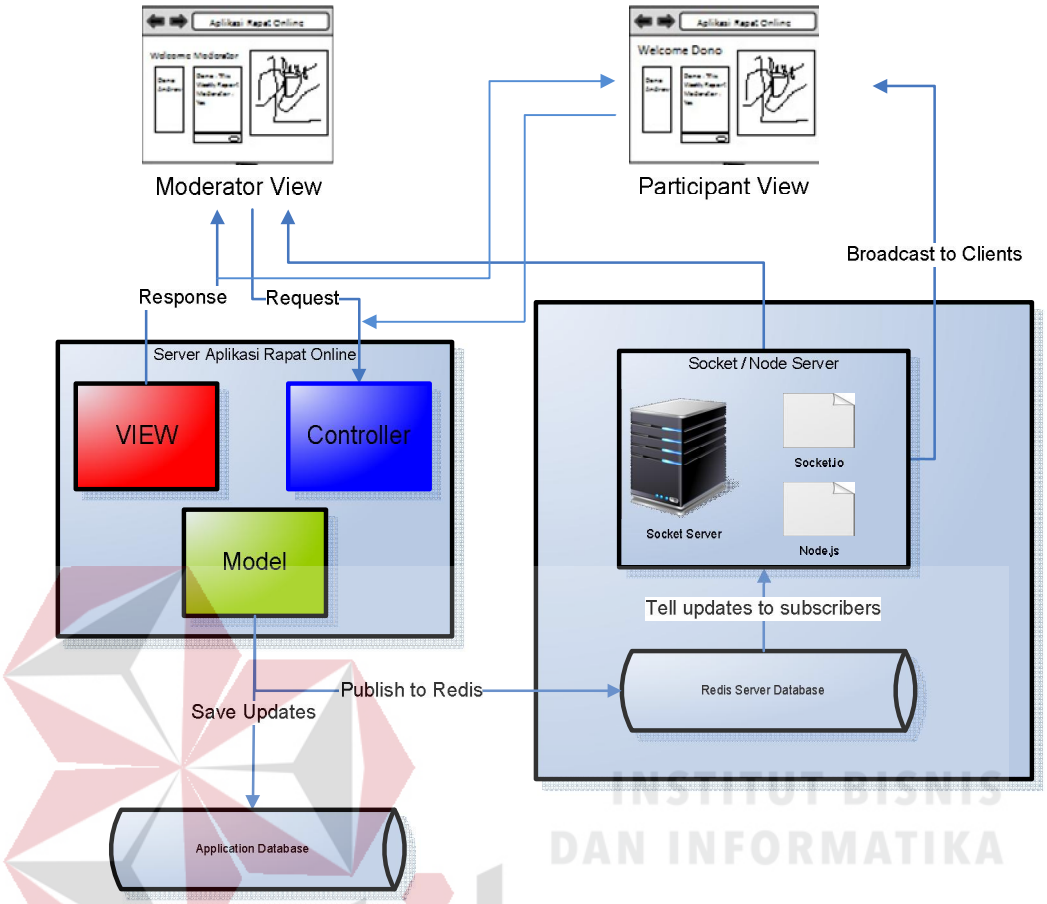
2. Staf Perusahaan / Peserta

Peserta rapat adalah para staf perusahaan yang terlibat di dalam proses rapat. Fasilitas seperti *chat*, voting, *sharing document*, serta berbagi *brainstorming board* dapat dilakukan, hanya saja kegiatan tersebut dibatasi oleh moderator.

3.3.1 Model Pengembangan Sistem

Pada model pengembangan sistem ini dimulai dengan mengumpulkan beberapa data yang digunakan sebagai *input*-an dari aplikasi. Data data tersebut meliputi Data *project*, data *sprint* dari *setiap project*, data peserta *project*, data *backlog* dan *file* penunjang rapat, yang selanjutnya akan diproses untuk menghasilkan informasi sesuai dengan tujuan pengembangan aplikasi.

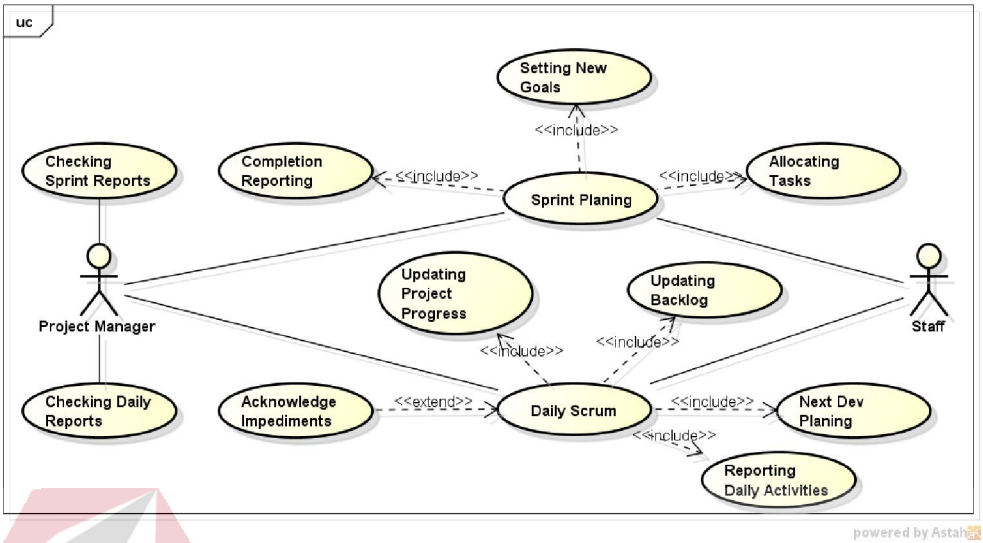
Pada pengembangan aplikasi rapat online ini, memanfaatkan teknologi *socket programming* yang diimplementasikan oleh *socket.io* dan *node*.js. Untuk menyimpan LOG dan History sementara sebelum melakukan *broadcast* ke setiap *client*, media penyimpanan yang digunakan adalah Database NOSQL, Redis. Sedangkan untuk pengembangan aplikasi utama, menggunakan *design pattern* MVC (Model-View-Controller). Menurut Firdaus (2008:2), MVC adalah merupakan pola pada pemrograman yang digunakan untuk memisahkan *data acces* dan *bussines logic* dari *data presentasion* dan *user interaction*.



Gambar 3.6 Arsitektur aplikasi rapat online

Pemisahan yang digambarkan pada Gambar 3.6 tersebut dilakukan dengan tujuan agar setiap perubahan yang terjadi pada *presentation logic* atau *bussines logic* tidak memberikan pengaruh satu sama lainya yang kompleks. Arsitektur MVC memisahkan aplikasi menjadi 3 bagian yaitu *model, view, controller*. Model merupakan representasi dari *database* dengan fungsi utama untuk menangani data, mengambil data, dan memanipulasi *database*. *View display* dari aplikasi yang merender data dari *model* dan kemudian dikirimkan ke *controller*. *Controller* merupakan perantara yang mendefinisikan perilaku aplikasi yang terjadi pada aplikasi, kemudian memetakannya menjadi aksi dari user terhadap model dan kemudian direspon oleh *view*.

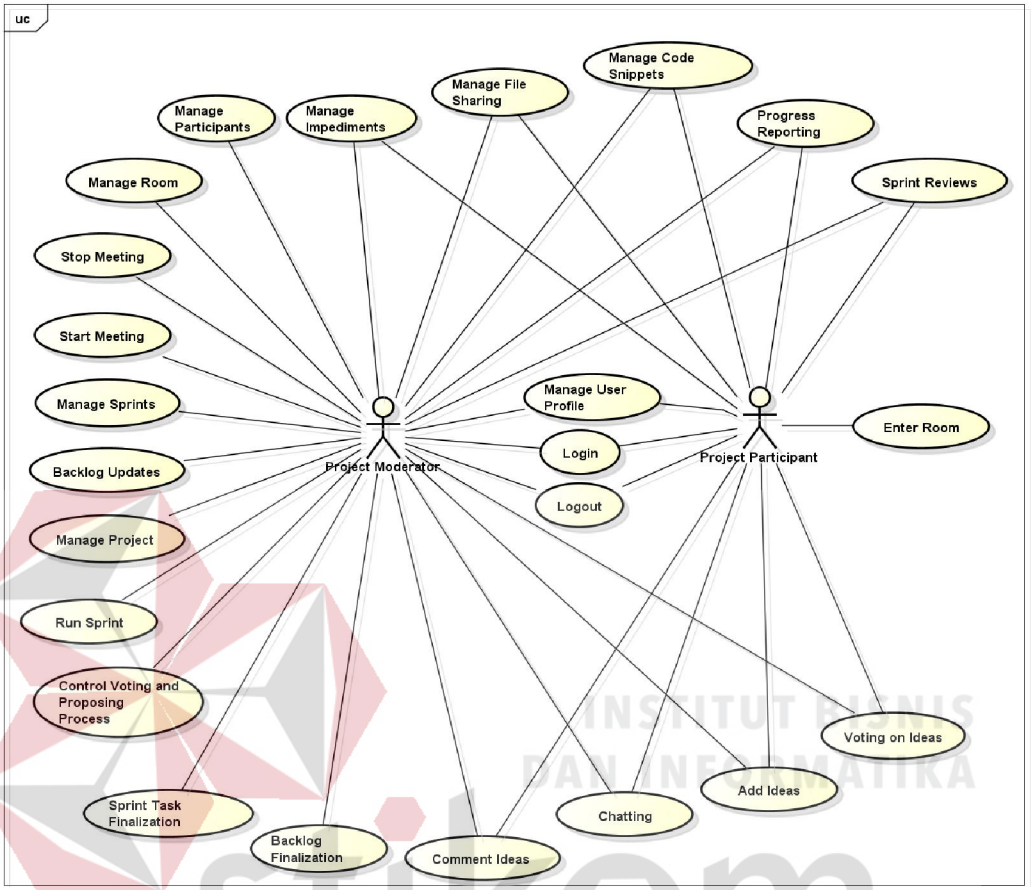
3.3.2 Use Case Diagram Bisnis Rapat pada PT. Garasilabs Manivesta



Gambar 3.7 *Use Case Diagram* Bisnis Rapat pada PT. Garasilabs Manivesta

Pada gambar 3.7 terdapat 2 aktor pengguna dalam proses bisnis yang ada yaitu *Project Manager* dan staf. Pada *use case* tersebut dijelaskan bahwa *manager* bertugas melihat laporan harian, proses pengembangan perangkat lunak (*by project*), serta melakukan kegiatan-kegiatan yang ada dalam rapat. Staf juga dapat melakukan kegiatan-kegiatan rapat seperti layaknya *manager*. Terdapat 2 proses penting dalam *scrum meeting,*yaitu *sprint planning* dan *daily scrum*. Dalam *Sprint planning*, ditentukan beberapa target yang dikerjakan dalam periode *sprint* serta melakukan alokasi tugas pada *sprint* tersebut.Pada waktu *sprint* diakhiri, juga disertai dengan pelaporan *progress* pengerjaan tugas – tugas yang telah dialokasikan sebelumnya. Sedangkan di dalam periode *sprint*, setiap harinya akan dilaksanakan *daily scrum*, yang akan berakhir pada akhir periode *sprint*. *Daily Scrum* berguna untuk melakukan pelaporan pengerjaan tugas – tugas dan juga melakukan kolaborasi lainnya seperti pemeahan masalah bersama.

3.3.3 Use Case Diagram Sistem Aplikasi Rapat Online



Gambar 3.8 *Use Case Diagram Sistem* Aplikasi Rapat Online

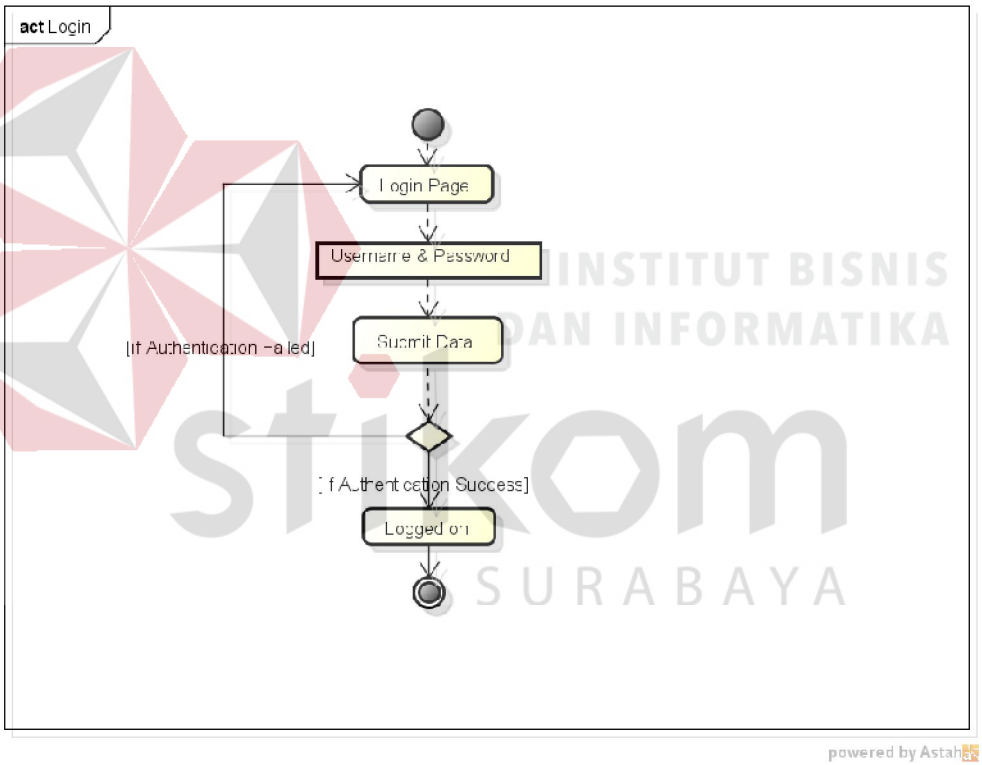
Pada gambar 3.8 terdapat 2 aktor pengguna dalam aplikasi yang akan dikembangkan yaitu *Project Moderator dan Project Participant*. Pada usecase tersebut dijelaskan secara teknis terhadap aplikasi, *Project Moderator* dapat melakukan fungsi-fungsi moderasi yang merupakan kontrol di dalam jalannya suatu rapat, melakukan finalisasi hasil rapat yang berupa *backlogs*, *sprint*, dan *tasks*, serta juga dapat berlaku sebagai staf. Sedangkan peserta hanya dapat melakukan fungsi-fungsi yang ada di dalam rapat kecuali fungsi moderasi. Usecase ini juga menerangkan proses-proses penunjang yang nantinya akan digunakan sebagai fitur dari aplikasi ini, seperti *Impediments*, *Code Snippets*, dan *Issue*.

3.3.4 *Activity Diagram*

Pada usecase diagram terdapat *activity diagrams* yang digunakan untuk menggambarkan aktivitas yang dilakukan pada tiap *use case*. Berikut ini adalah *activity diagram* dari *use case* yang telah dibuat .

A. Activity Diagram *use case* login

Activity Diagram use case login menggambarkan aktivitas yang terjadi pada proses login.

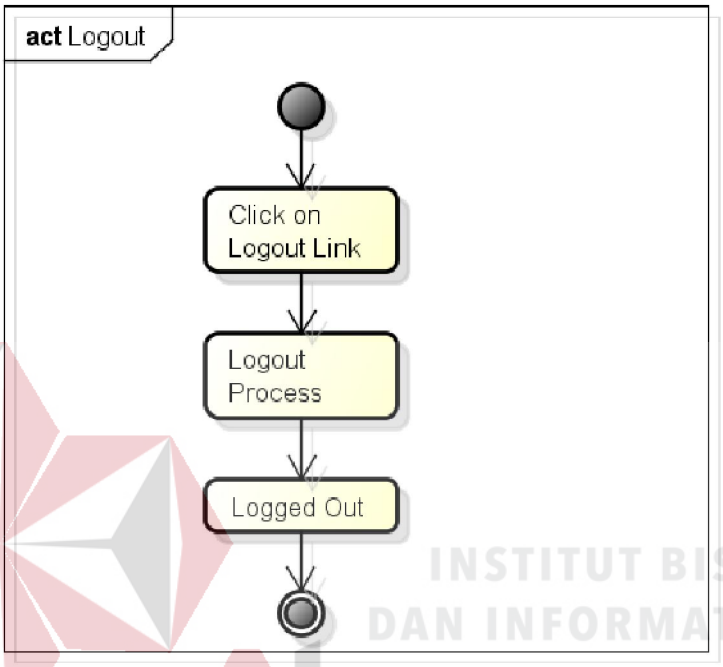


Gambar 3.9 *Activity Diagram use case* login

Pada *activity diagram* seperti Gambar 3.9, dijelaskan alur autentikasi untuk pengguna sebelum menggunakan aplikasi ini. *Username* dan *Password* dimasukkan pada halaman *login* dan dilakukan pengecekan untuk memeriksa apakah pengguna tersebut ada ataupun tidak. Jika autentikasi tidak berhasil, maka pengguna akan diarahkan kembali ke dalam halaman *login*.

B. Activity Diagram *use case* logout

Activity Diagram use case logout menggambarkan aktivitas yang terjadi pada proses logout.

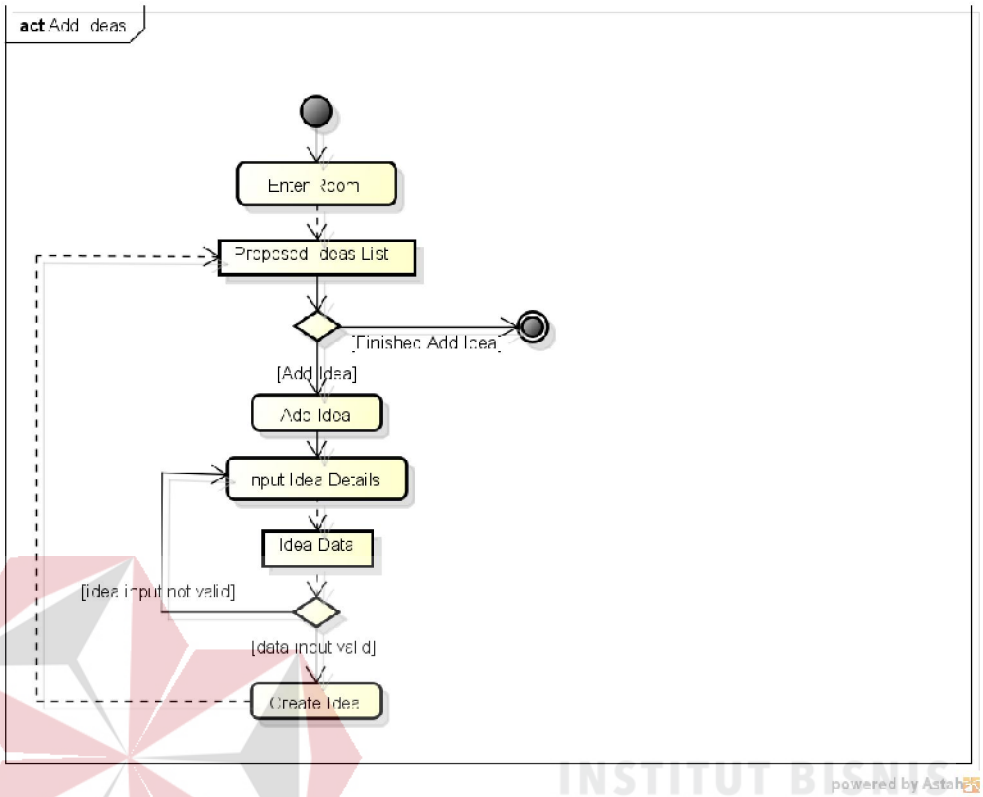


Gambar 3.10 *Activity Diagram use case* logout

Pada *activity diagram* seperti Gambar 3.10, dijelaskan alur bagi pengguna untuk keluar dari aplikasi. Pada waktu pengguna mengklik tombol *sign out*, maka *user session* akan dihapus, sehingga status pengguna tidak lagi masuk di dalam aplikasi.

C. Activity Diagram *use case* add ideas

Activity Diagram use case add ideas menggambarkan aktivitas yang dilakukan pengguna untuk menambahkan ide.

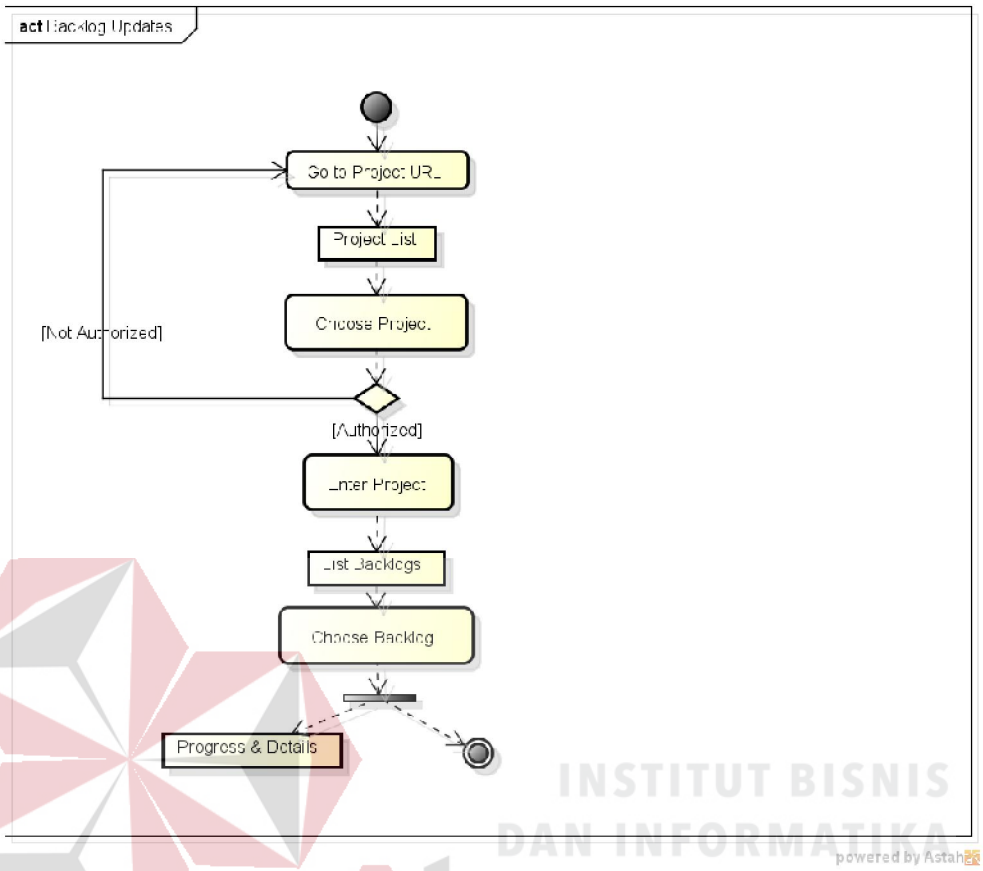


Gambar 3.11 *Activity Diagram use case* Add Idea

Pada *activity diagram* seperti Gambar 3.9, dijelaskan alur pengguna dalam memberikan ide. Aktivitas ini terjadi di dalam ruang rapat. Pengguna akan mengajukan ide individu beserta detail berupa penjelasan yang mendukung ide tersebut. Terdapat autentikasi terhadap ide yang dimasukkan, sehingga jika ide yang diajukan tidak lolos autentikasi, maka ide tidak dapat disimpan.Pemberian ide terdapat pada ruangan rapat pada tahap *sprint planning*, karena pada *sprint planning*, akan ditempatkan fitur – fitur atau tugas – tugas baru yang akan dikerjakan pada periode *sprint* yang akan datang.

D. Activity Diagram *use case* backlog updates

Activity Diagram use case backlog updates menggambarkan aktivitas yang dilakukan pengguna untuk melihat perubahan dari backlogs.

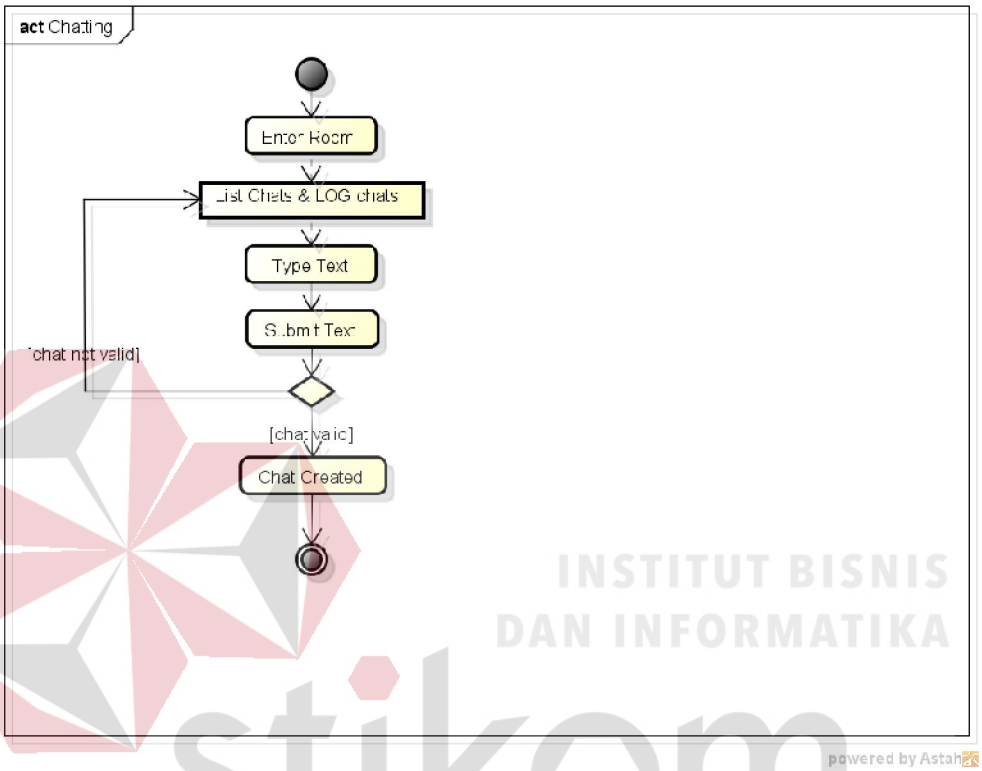


Gambar 3.12 *Activity Diagram use case* Backlog Updates

Pada *activity diagram* seperti Gambar 3.12, dijelaskan alur pengguna dalam melihat tahap pengerjaan dari setiap tugas pada suatu *sprint* pada setiap *project*. Setiap pengguna hanya dapat melihat *progress* pengerjaan pada setiap *project* dimana pengguna tersebut terdaftar didalamnya.

E. Activity Diagram *use case* chatting

Activity Diagram use case chatting menggambarkan aktivitas yang dilakukan pengguna untuk melakukan interaksi *chatting*.

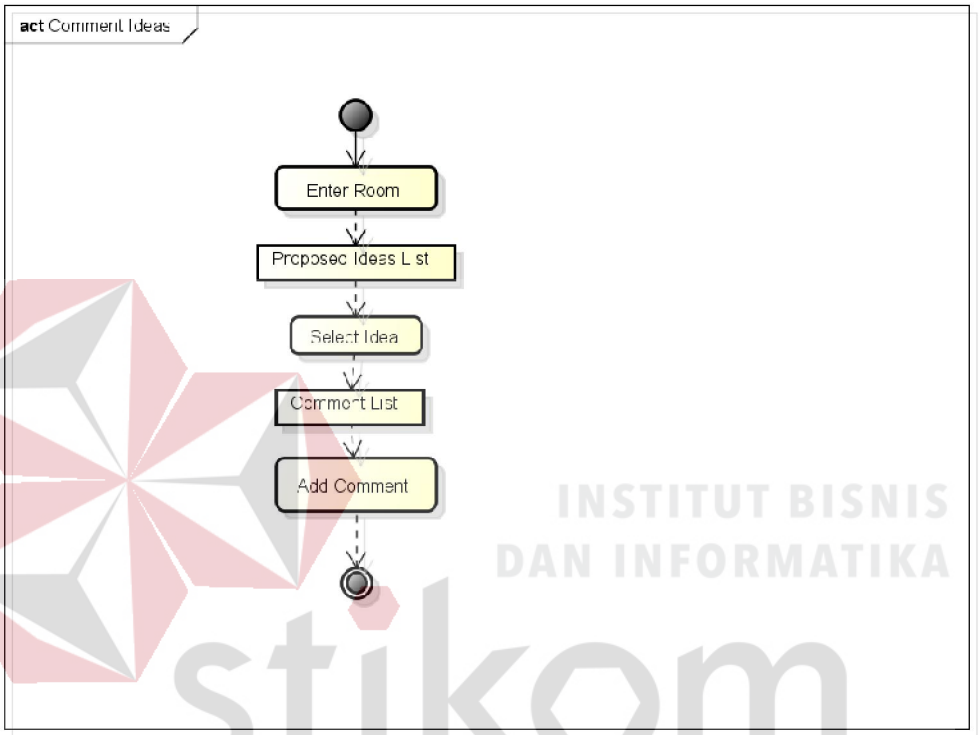


Gambar 3.13 *Activity Diagram use case* Chatting

Pada *activity diagram* seperti Gambar 3.13, dijelaskan fitur *chatting* pada aplikasi rapat online ini. Fitur ini merupakan media komunikasi umum pada aplikasi ini. Terdapat validasi pada setiap interaksi yang dilakukan, sehingga akan dilakukan pengecekan setiap *chat* yang dilakukan tidak dapat kosong. Pada waktu pengguna baru saja memasuki ruangan, akan ditampilkan *history chat* yang telah terjadi.

F. Activity Diagram *use case* comment ideas

Activity Diagram use case comment ideas, menggambarkan aktivitas pengguna di dalam proses menambahkan komentar pada setiap ide yang telah dibuat di dalam ruangan.

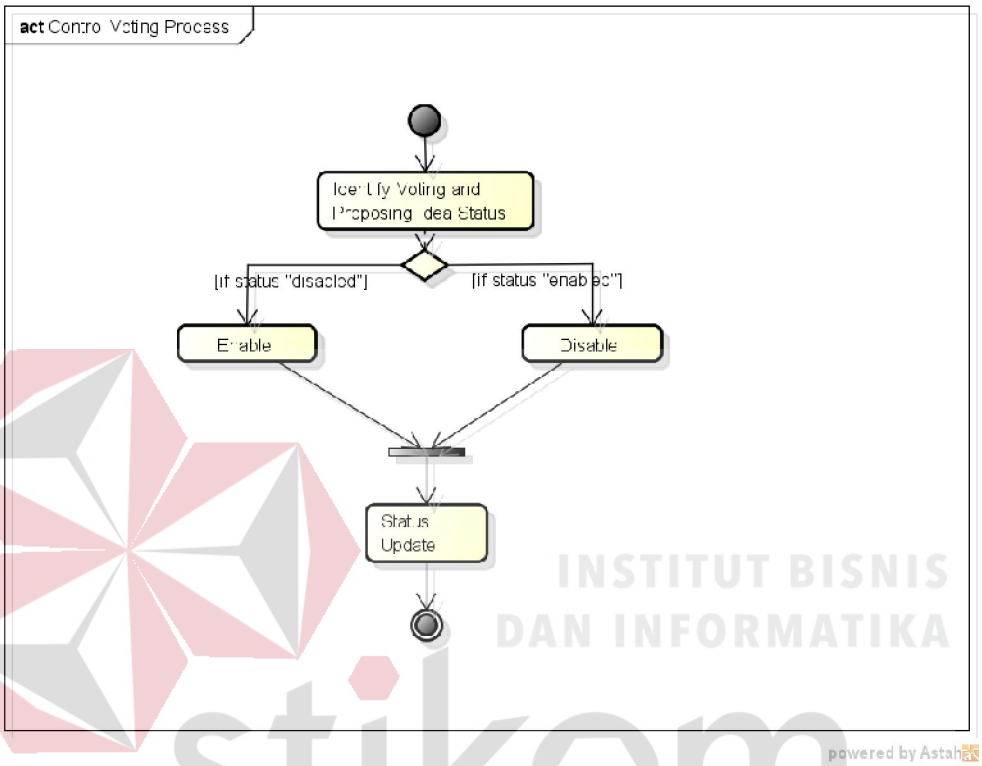


Gambar 3.14 *Activity Diagram use case* Comment Ideas

Activity diagram seperti Gambar 3.14, menjelaskan aktivitas pengguna untuk memasukkan komentar pada setiap ide yang telah diajukan oleh peserta rapat. Pengguna hanya memasukkan komentar berupa narasi dan akan ditampilkan pada halaman detil pada setiap ide

G. Activity Diagram *use case* control voting and proposing process

Activity Diagram ini menggambarkan aktivitas yang dilakukan moderator dalam melakukan kontrol pada ruangan, saat proses voting dan pengajuan ide-ide.

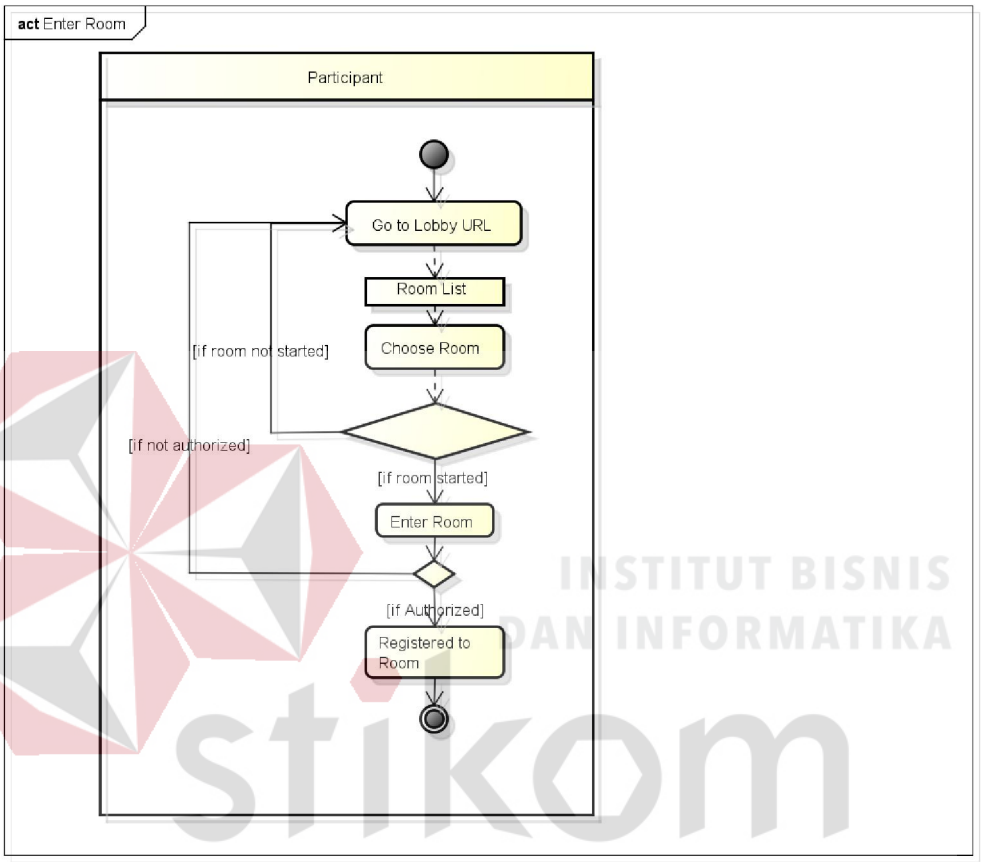


Gambar 3.15 *Activity Diagram use case* Control Voting Process

Activity diagram seperti Gambar 3.15, menjelaskan aktivitas moderator untuk melakukan limitasi pemberian ide oleh peserta rapat. Status *voting* akan secara otomatis di-*set* "*disabled*", lalu moderator dapat mengubah status *voting* menjadi "*enabled*".

H. Activity Diagram *use case* enter room

Activity Diagram ini menggambarkan aktivitas yang dilakukan peserta rapat untuk memasuki ruang rapat yang telah dimulai oleh moderator.

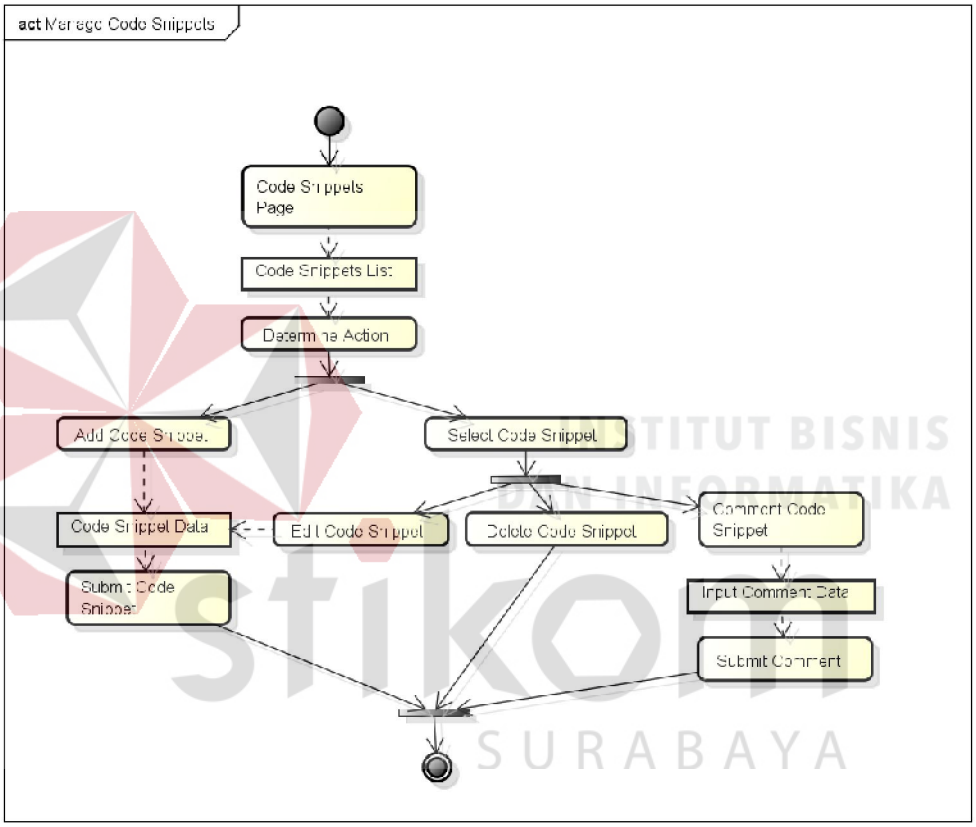


Gambar 3.16 *Activity Diagram use case* Control Enter Room

Activity diagram seperti Gambar 3.16, daftar ruangan rapat akan ditampilkan pada halaman *lobby*. Peserta memasuki salah satu ruangan untuk mengikuti rapat. Tetapi jika ruangan tersebut belum dimulai oleh moderator, maka akan ditampilkan notifikasi *error* dan pengguna akan diarahkan kembali ke halaman *lobby*.

I. Activity Diagram *use case* code snippets

Activity Diagram use case code snippets ini menggambarkan aktivitas yang dilakukan pengguna untuk menggunakan fasilitas snippets untuk melakukan dokumentasi *coding* yang akan dapat digunakan oleh pengguna lainnya pada waktu tertentu.

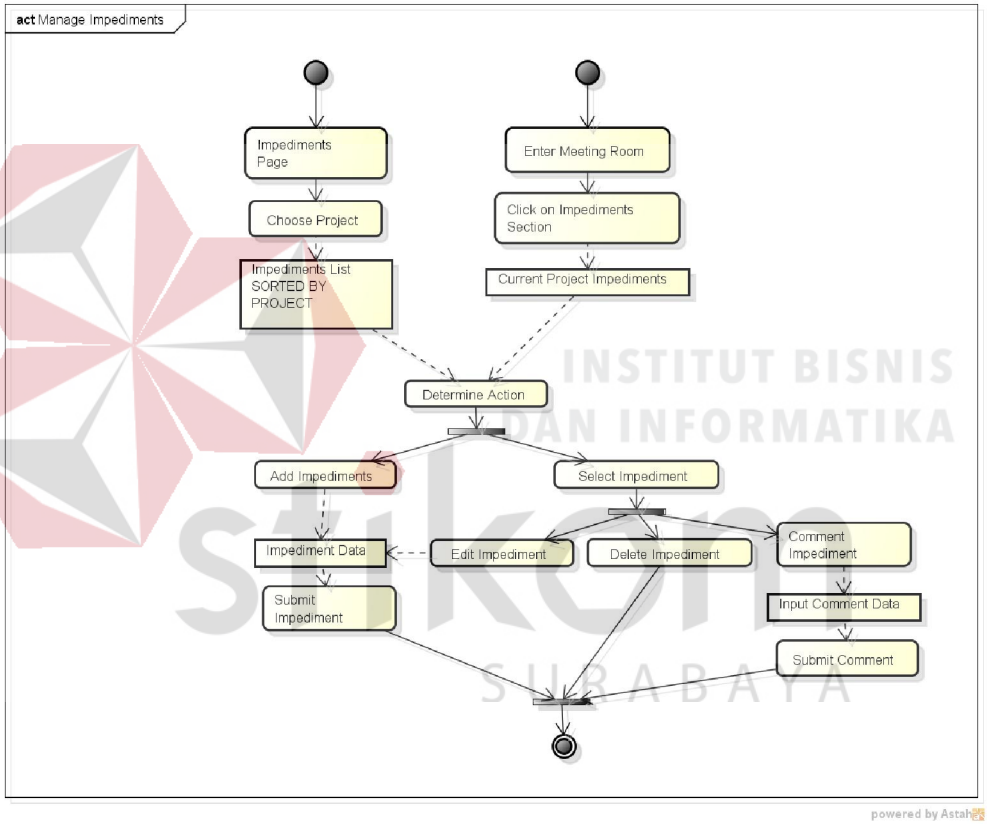


Gambar 3.17 *Activity Diagram use case* Manage Code Snippets

Activity diagram seperti Gambar 3.17 menjelaskan terdapat 3 aksi yang dapat dilakukan oleh pengguna pada setiap *code snippet* yang telah terbuat, seperti memberikan komentar, mengubah *coding*, serta menghapus *snippet*. Komentar dilakukan sebagai bentuk kolaborasi untuk bersama mengkoreksi suatu *coding*.

J. Activity Diagram *use case* manage impediments

Activity Diagram use case manage impediments menggambarkan aktivitas yang dilakukan pengguna untuk mngelola *impediments* dan untuk melakukan kolaborasi penyelesaian masalah. *Impediment* yang dimasukkan juga nantinya akan dapat dipergunakan sebagai dokumentasi atas permasalahan yang pernah terjadi.

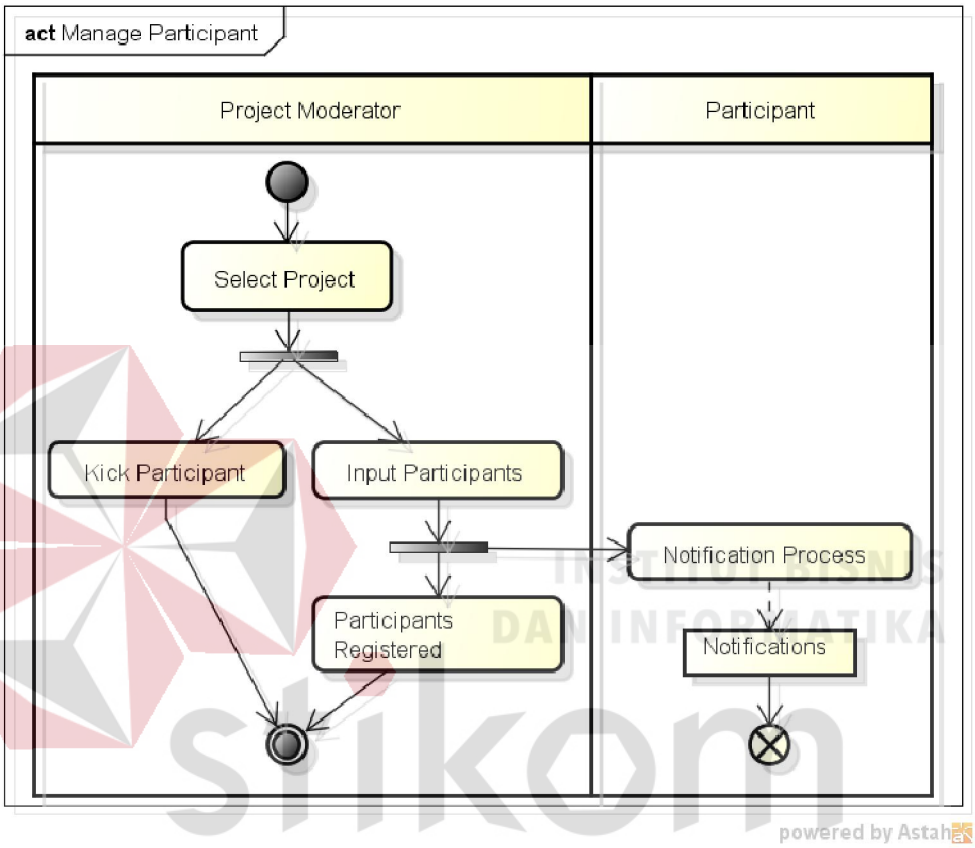


Gambar 3.18 *Activity Diagram use case* Manage Impediments

Activity diagram seperti Gambar 3.18 menjelaskan terdapat 3 aksi yang dapat dilakukan oleh pengguna pada setiap *impediment* yang telah terbuat, seperti memberikan komentar, mengubah status *impediment*, serta menghapus *impediment*. Komentar dilakukan sebagai bentuk kolaborasi untuk bersama memecahkan kesulitan (*impediment*) dari seorang peserta *project*.

K. Activity Diagram *use case* manage participants

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola peserta rapat.

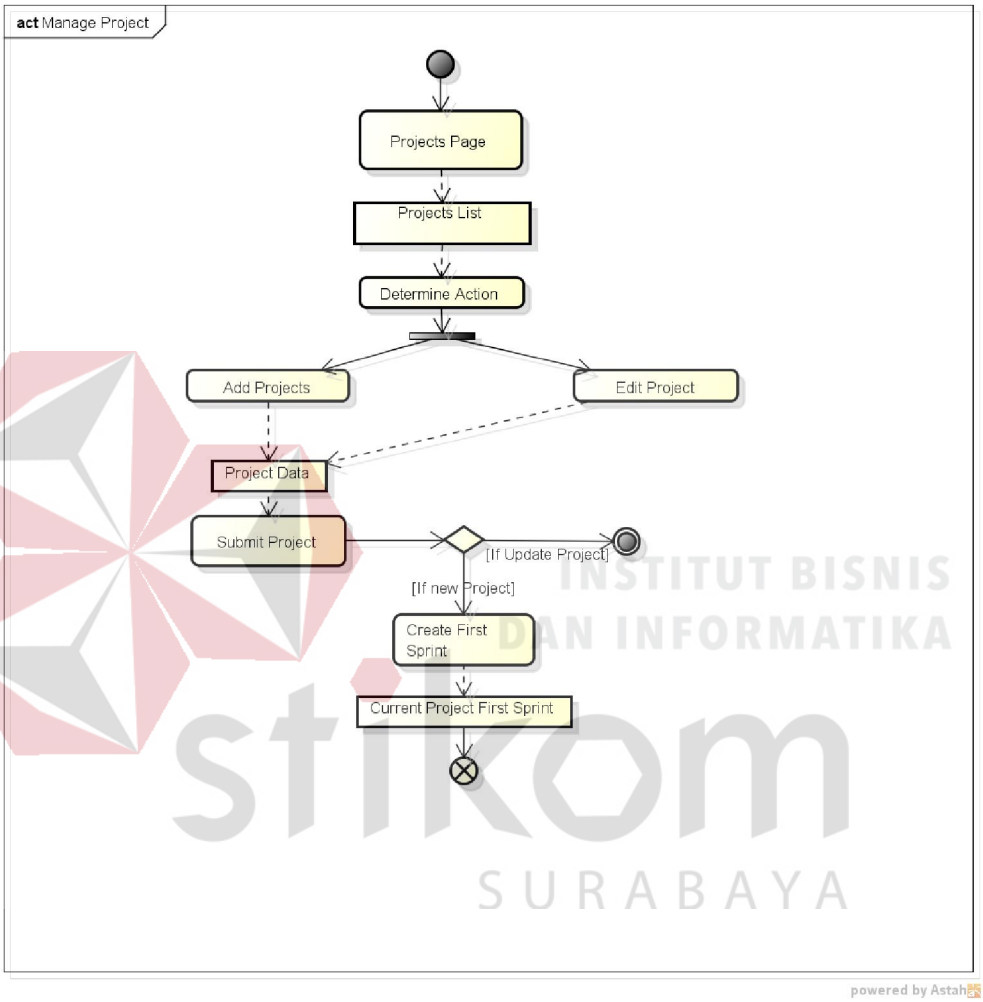


Gambar 3.19 *Activity Diagram use case* Manage Participant

Activity diagram seperti Gambar 3.19 menjelaskan aktivitas moderator *project* untuk melakukan penambahan ataupun pengurangan peserta *project*. Pada waktu moderator memasukkan salah satu pengguna kedalam suatu *project*, maka akan terdapat notifikasi yang akan terkirim kepada pengguna tersebut.

L. Activity Diagram *use case* manage project

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola *project*.

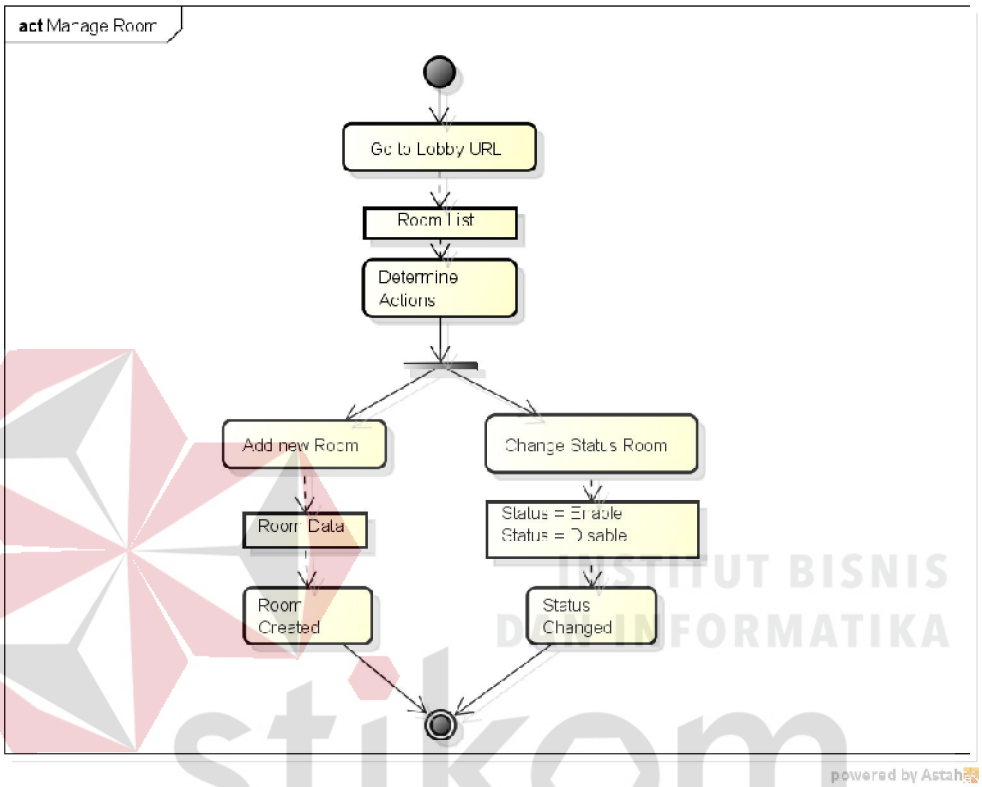


Gambar 3.20 *Activity Diagram use case* Manage Project

Activity diagram seperti Gambar 3.20 pengelolaan *project* hanya terbatas pada penambahan dan pengubahan detil *project*. Pada saat *project* pertama kali terbuat, *sprint* "*initialized*" akan secara otomatis terbuat oleh aplikasi, dengan status "*new sprint*".

M. Activity Diagram *use case* manage room

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola ruangan.

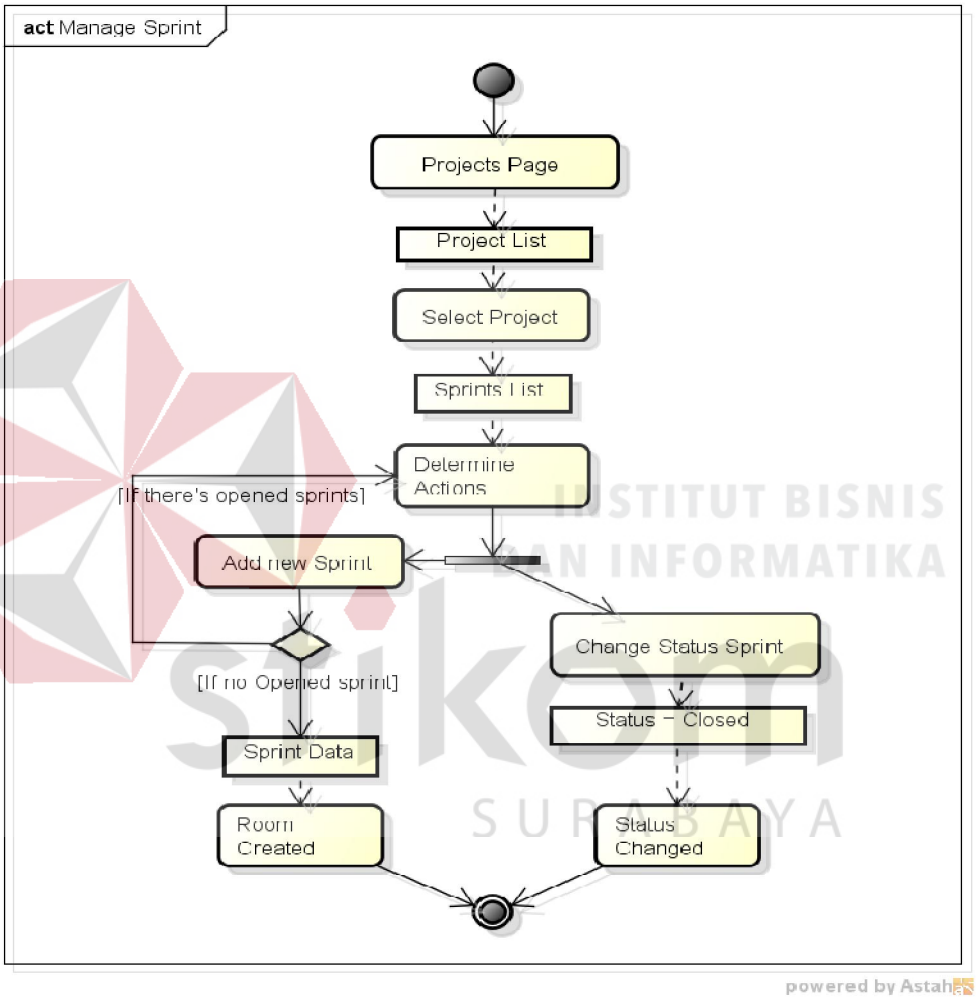


Gambar 3.21 *Activity Diagram use case* Manage Room

Activity diagram seperti Gambar 3.21 dijelaskan bahwa *project* moderator mempunyai kemampuan untuk memulai dan mengakhiri proses rapat yang terjadi pada suatu ruangan. Proses memulai dan mengakhiri proses rapat yang terjadi adalah dengan mengubah status dari ruangan menjadi *enable* atau *disable*. *Project Moderator* juga dapat membuat ruangan baru dari setiap *sprint* yang tersedia

N. Activity Diagram *use case* manage sprints

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola *sprint*. Serta juga menggambarkan aktivitas yang dilakukan oleh peserta rapat untuk me-*review sprint*.



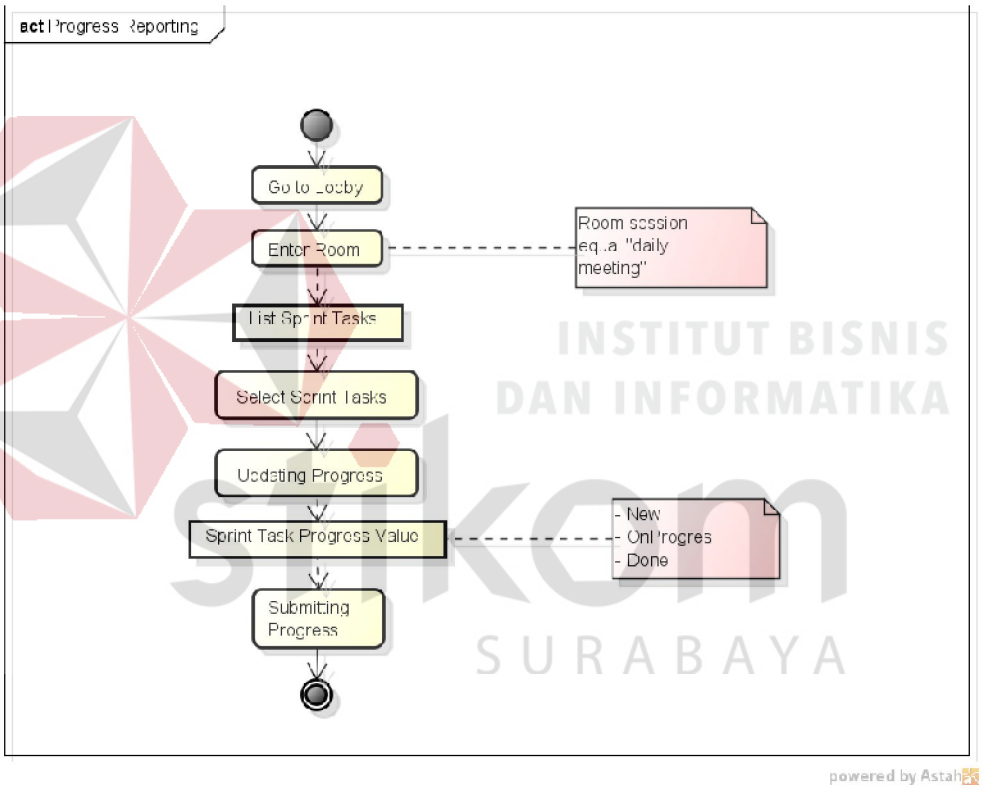
Gambar 3.22 *Activity Diagram use case* Manage Sprints

Activity diagram seperti Gambar 3.22 menjelaskan aktivitas moderator *project* untuk melakukan pengelolaan terhadap *sprint*. Moderator dapat mengubah status *sprint* menjadi *closed* jika periode *sprint* telah berakhir. Setelah mengubah

status *sprint*, maka moderator dapat menambah *sprint* baru untuk periode pengembangan selanjutnya.

O. Activity Diagram *use case* progress reporting

Activity Diagram ini menggambarkan aktivitas yang dilakukan pengguna dalam melakukan pelaporan perkembangan pengembangan perangkat lunak pada setiap *sprint*.

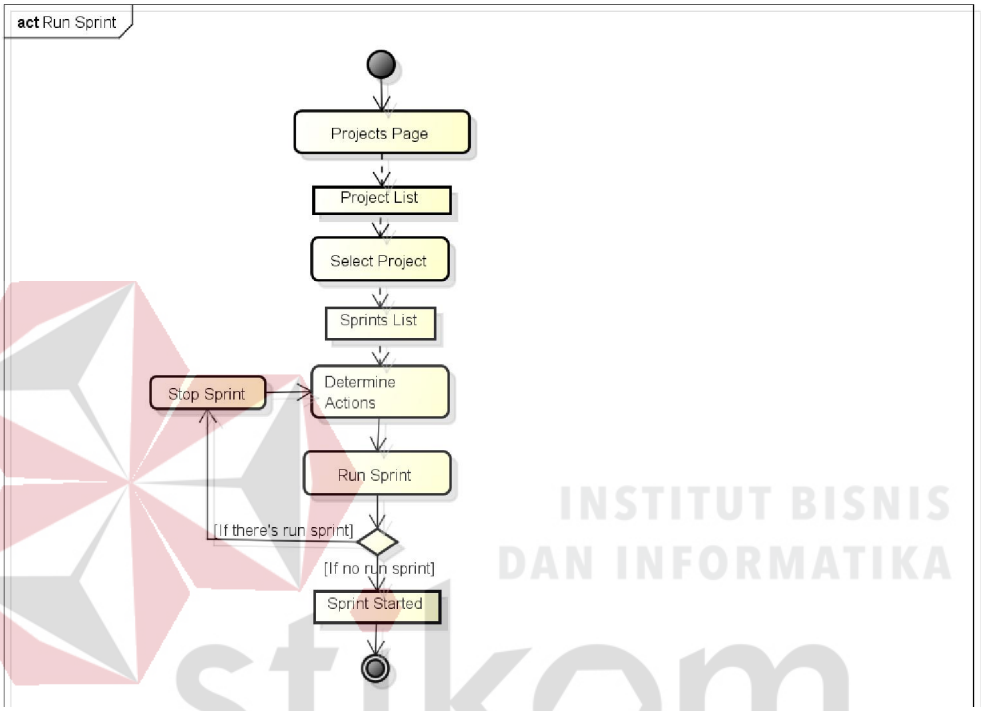


Gambar 3.23 *Activity Diagram use case* Progress Reporting

Activity diagram seperti Gambar 3.23 dijelaskan bahwa proses *reporting* tugas ini hanya dapat terjadi jika status ruangan rapat adalah "*daily meeting*", dengan kata lain, rapat dilakukan pada saat *sprint* sedang berjalan. *Update progress* setiap tugas bertujuan untuk mengubah status pengerjaan yang terdiri dari *new* "*on*", "*progress*", dan "*done*".

P. Activity Diagram *use case* run sprint

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk menjalankan *sprint*. Dengan menjalankan *sprint*, maka rapat dari *sprint* dapat dilakukan.

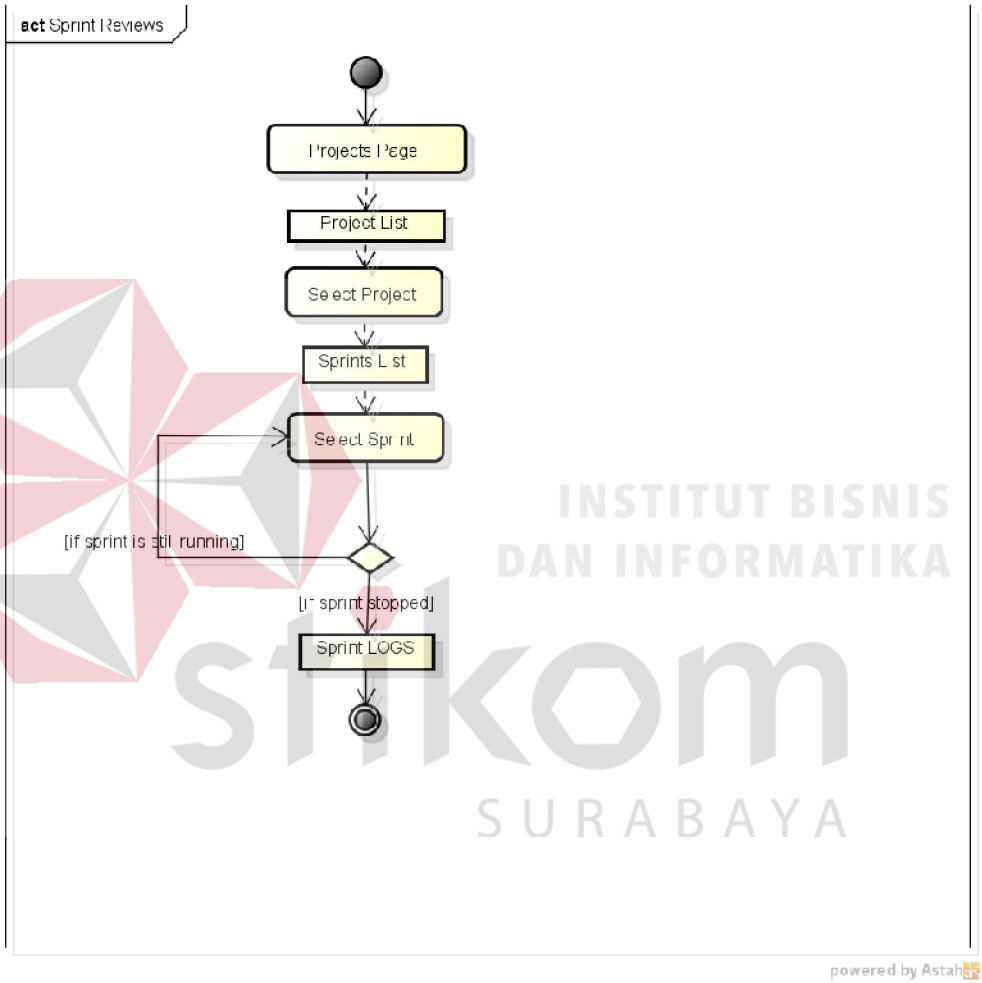


Gambar 3.24 *Activity Diagram use case* Run Sprint

Activity diagram seperti Gambar 3.24 menjelaskan aktivitas moderator *project* untuk memulai *sprint* sebagai periode pengembangan berikutnya. Pada setiap *project*, hanya terdapat satu *sprint* yang berjalan.

Q. Activity Diagram *use case* sprint reviews

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk melakukan *review* terhadap *sprint*, riwayat aktivitas dan penyelesaian di dalam *sprint* juga ditampilkan.

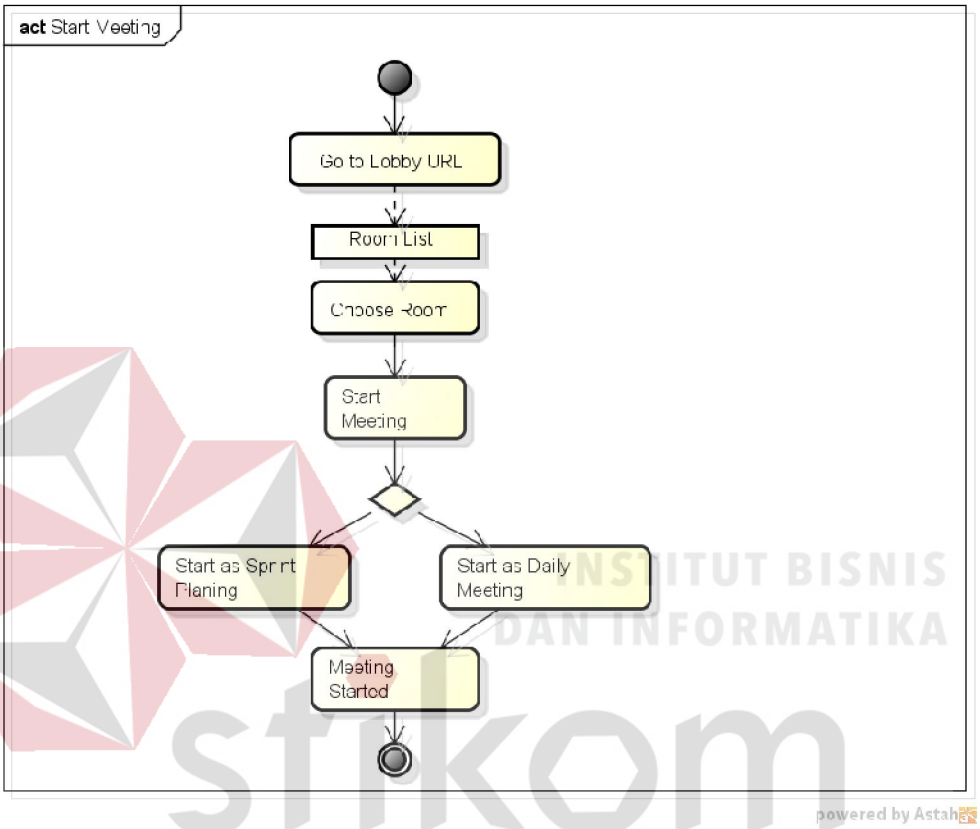


Gambar 3.25 *Activity Diagram use case* Sprint Reviews

Activity diagram seperti Gambar 3.25 menjelaskan aktivitas moderator *project* untuk melakukan pengawasan setiap *sprint*. Pada setiap *sprint* akan terdapat *history* aktivitas yang dilakukan pada setiap rapat pada *sprint* tersebut.

R. Activity Diagram *use case* start meeting

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk memulai rapat.

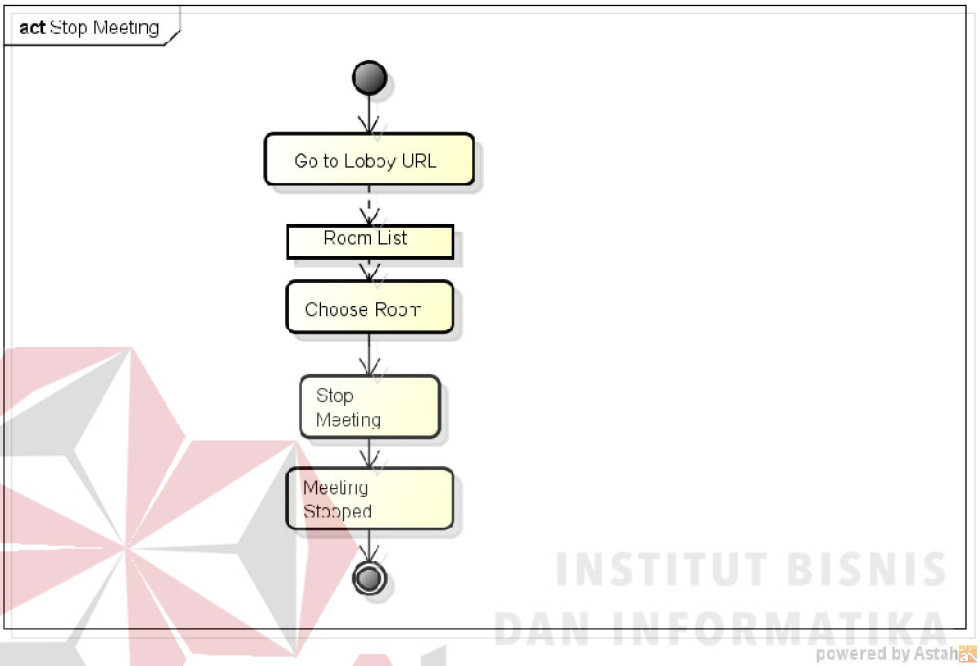


Gambar 3.26 *Activity Diagram use case* Start Meeting

Pada Gambar 3.26, dijelaskan bahwa dalam memulai rapat, *project moderator* dapat hanya melakukan klik pada link "*run meeting*". Setelah itu akan terjadi pengecekan status *sprint*, jika *sprint* telah berjalan maka status rapat adalah *daily meeting*, tetapi jika *sprint* belum berjalan maka status rapat adalah *sprint planning*.

S. Activity Diagram *use case* stop meeting

Activity Diagram ini menggambarkan aktivitas yang dilakukan *project moderator* untuk menghentikan rapat.

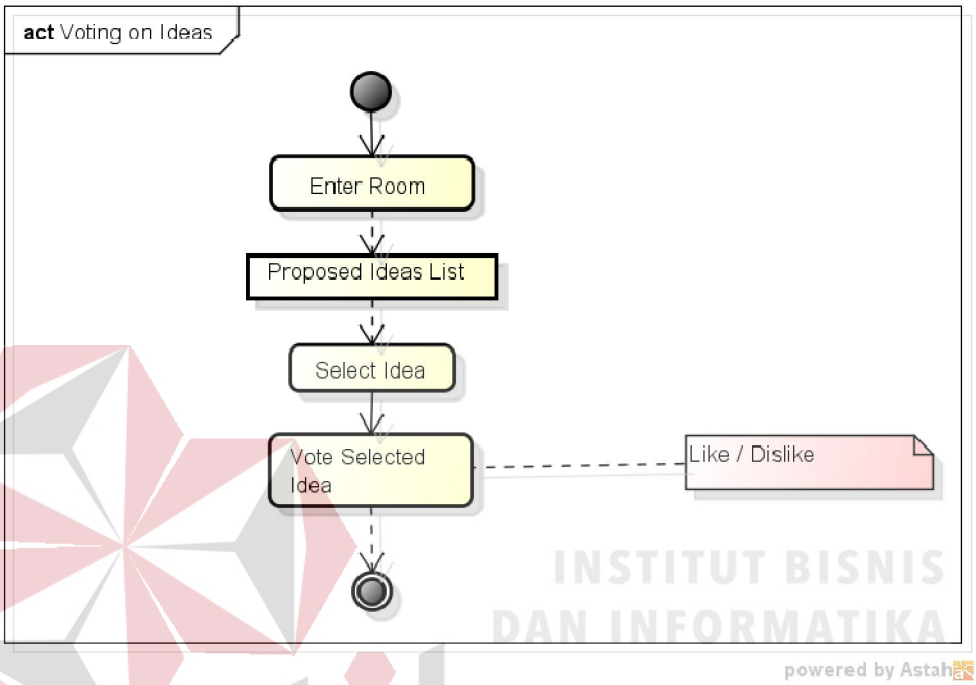


Gambar 3.27 *Activity Diagram use case* Stop Meeting

Pada Gambar 3.27, dijelaskan aktivitas menghentikan rapat, *project moderator* dapat hanya melakukan klik pada link "*stop meeting*" dan akan mengubah status rapat menjadi "*inactive*". Sehingga peserta rapat tidak dapat memasuki ruangan rapat.

T. Activity Diagram *use case* voting on ideas

Activity Diagram ini menggambarkan aktivitas yang dilakukan pengguna dalam melakukan voting untuk setiap ide yang ada.

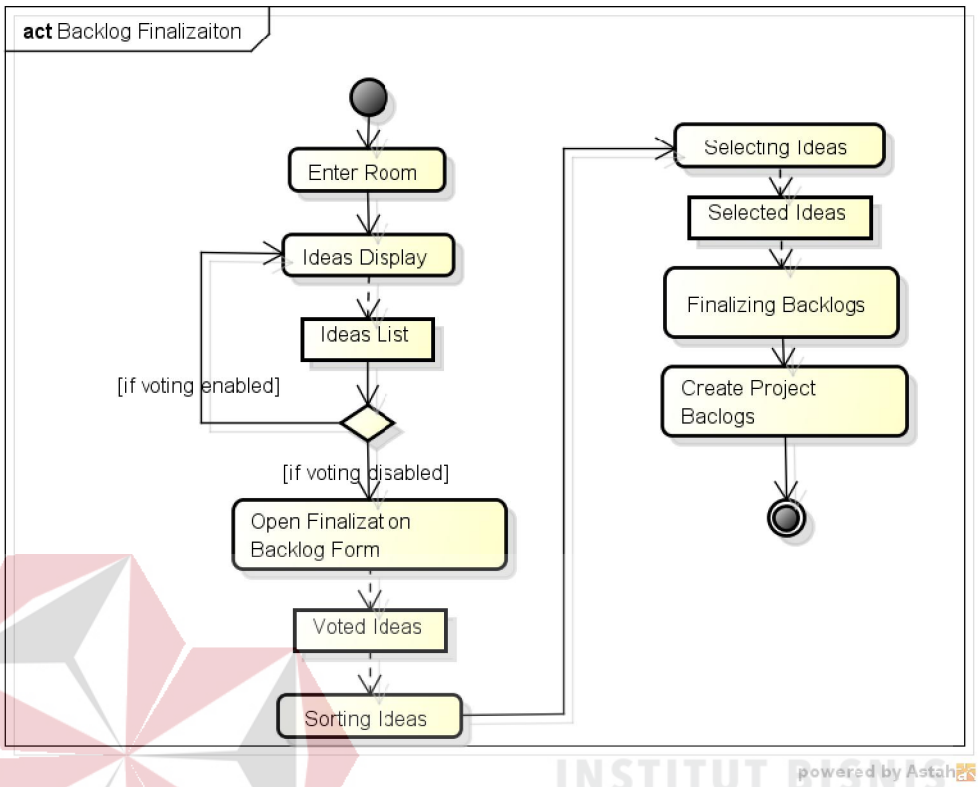


Gambar 3.28 *Activity Diagram use case* Voting on ideas

Pada Gambar 3.28, dijelaskan proses melakukan *voting* pada ruangan rapat. *Voting* dilakukan pada setiap ide yang diajukan oleh peserta rapat. *Voting* akan berupa status "*like*" atau "*dislike*". *Voting* terdapat pada *sprint planning*. Hasil dari voting dapat digunakan untuk me

U. Activity Diagram *use case* backlog finalization

Activity Diagram ini menggambarkan aktivitas yang dilakukan moderator dalam melakukan finalisasi *backlogs* dari setiap ide yang ada.

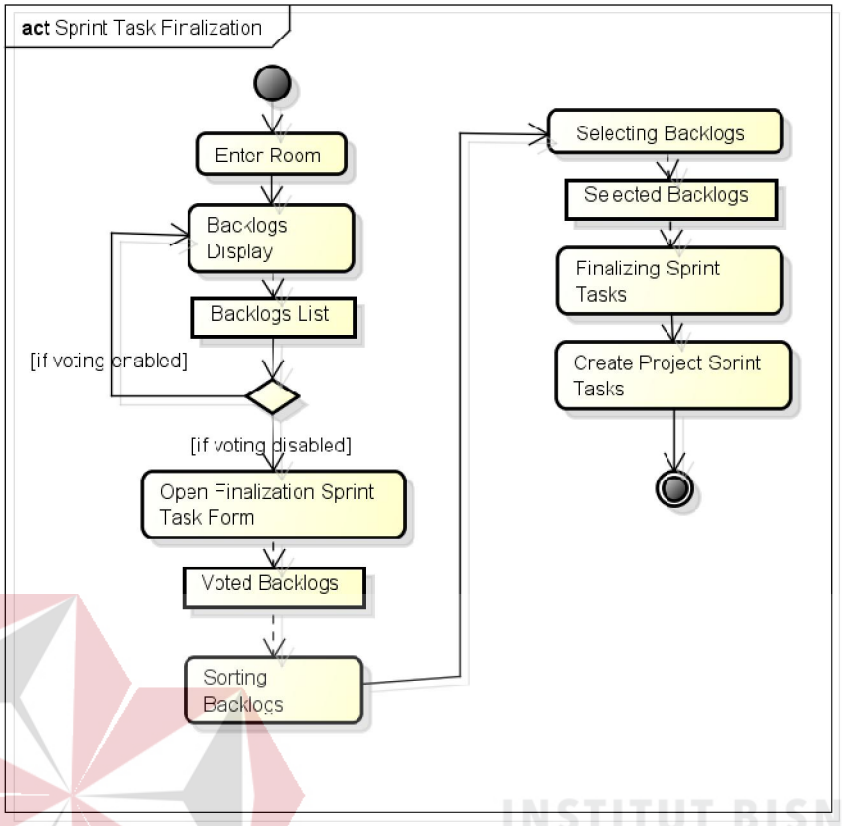


Gambar 3.29 *Activity Diagram use case* Backlog Finalization

Pada Gambar 3.29, dijelaskan aktivitas yang dilakukan oleh moderator *project* untuk melakukan seleksi dari ide-ide yang telah diajukan oleh setiap peserta rapat, menjadi *backlog project*. Ide yang telah diajukan, diberi komentar, dan di-*voting*, akan diurutkan oleh moderator dan dipilih satu persatu untuk menjadi *backlog candidate*. Setelah selesai melakukan pemilihan, maka akan dilakukan finalisasi *backlog candidates*, sehingga resmi menjadi *project backlogs*.

V. Activity Diagram *use case* Sprint Task Finalization

Activity Diagram ini menggambarkan aktivitas lanjutan dari *backlog finalization* yang dilakukan moderator, dengan tujuan melakukan finalisasi *sprint tasks* dari setiap *backlog* yang ada.

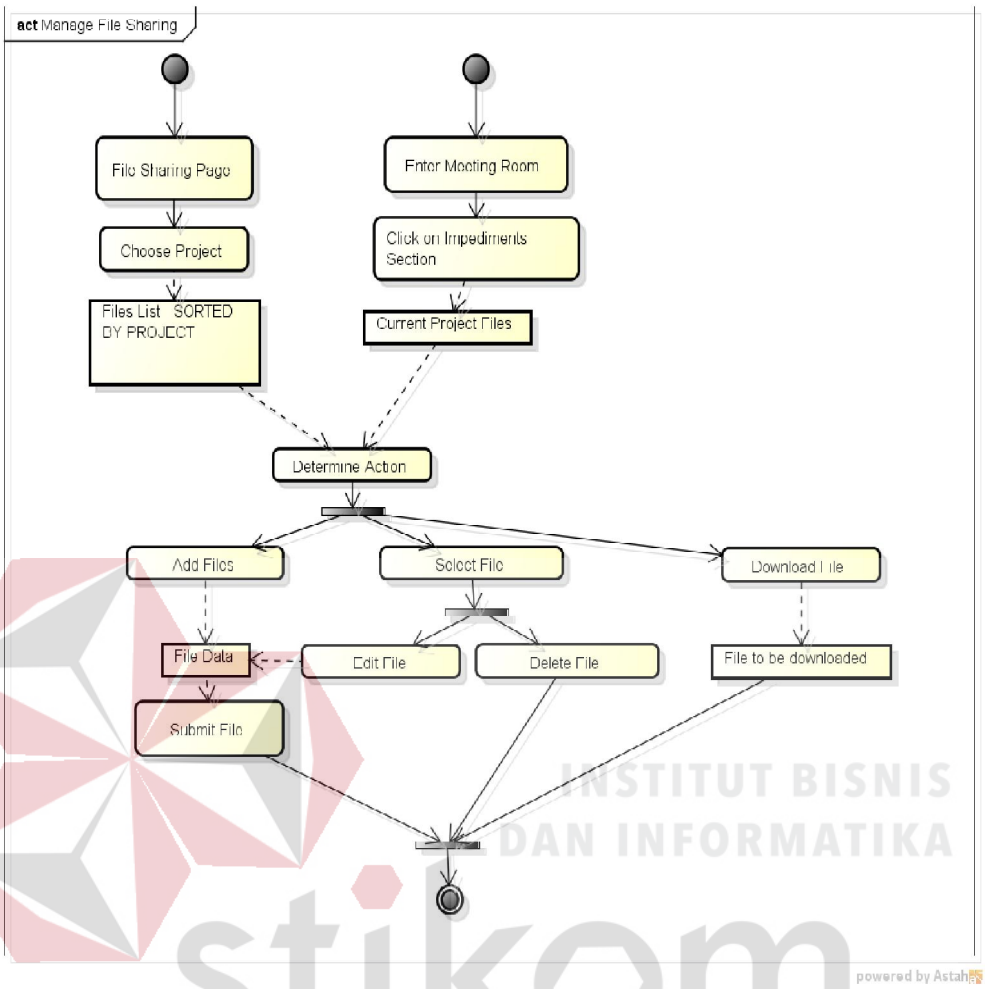


Gambar 3.30 *Activity Diagram use case* Sprint Task Finalization

Pada Gambar 3.30, dijelaskan aktivitas yang dilakukan oleh moderator *project* untuk melakukan finalisasi pada *sprint tasks* yang dipilih dari *project backlog* ada. Jadi pada saat finalisaasi, *sprint tasks* yang dipilih akan dimasukkan menjadi *tasks* dari *sprint* yang ada dan siap untuk dikerjakan.

W. Activity Diagram *use case* manage file sharing

Activity Diagram ini menggambarkan aktivitas yang dilakukan pengguna untuk melakukan *file sharing*.

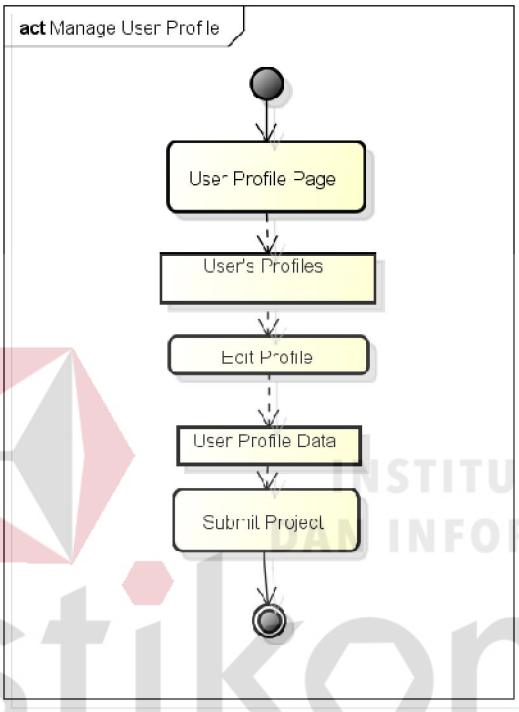


Gambar 3.31 *Activity Diagram use case* Manage File Sharing

Pada Gambar 3.31, dijelaskan aktivitas yang dilakukan oleh moderator *project* untuk melakukan finalisasi pada *sprint tasks* yang dipilih dari *project backlog* ada. Jadi pada saat finalisaasi, *sprint tasks* yang dipilih akan dimasukkan menjadi *tasks* dari *sprint* yang ada dan siap untuk dikerjakan.

X. Activity Diagram *use case* manage user profile

Pada Gambar 3.32, dijelaskan aktivitas yang dilakukan oleh pengguna aplikasi untuk mengubah detil dari data diri pengguna seperti *password* dan *email* pengguna.



Gambar 3.32 *Activity Diagram use case* Manage User Profile

3.3.5 *Flow of Event*

Pada usecase diagram terdapat *flow of event* yang digunakan untuk menggambarkan aktivitas yang dilakukan pada tiap *use case*. Berikut ini adalah *flow of event* dari *use case* yang telah dibuat .

A. Flow of Event *use case* login

Flow of event use case login menggambarkan aktivitas yang terjadi pada proses login. Terdapat beberapa proses antara lain melakukan input *username* dan *password*, melakukan pengecekan pengguna, validasi *input*-an pengguna seperti terlihat pada tabel berikut ini.

Tabel 3.2 *Flow of Event use case* login

Usecase Login		
Nama	Login	
Deskripsi Singkat	End-User akan menggunakan Fitur Login untuk memakai aplikai	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan registrasi terlebih dahulu	
Alur Utama	1	End-User memasukkan Email dan Password
	2	Aplikasi akan melakukan cek ke dalam database apakah Email terdaftar
	3	Aplikasi akan melakukan cek ke dalam database apakah Email yang terdaftar mempunyai Password yang telah diinput
	4	Aplikasi akan melakukan *set session "current_user"*
	5	End-User akan masuk ke dalam aplikasi
Alur Alternatif	A1	Aplikasi menampilkan kesalahan apabila End-User belum mengisi email dan atau password
	A2	Aplikasi menampilkan kesalahan apabila email dan atau password belum terdaftar
Kondisi Sukses		Pengguna masuk ke dalam aplikasi dan session "current_user" telah terdaftar

Tabel 3.2 menjelaskan alur proses login berdasarkan kejadian yang dialami oleh pengguna saat menjalankan proses login yang meliputi validasi kesalahan pemasukan *username* dan *password*, pemberitahuan yang muncul setelah validasi dijalankan, dan kejadian ketika proses sukses dijalankan.

B. Flow of event *use case* logout

Flow of Event use case logout menggambarkan aktivitas yang terjadi pada proses logout

Tabel 3.3 *Flow of Event use case* logout

Usecase Logout		
Nama	Logout	
Deskripsi Singkat	End-User akan menggunakan Fitur Logout untuk keluar dari aplikasi	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User melakukan klik pada tombol "sign out"
	2	Aplikasi akan melakukan proses logging out untuk End-User
Kondisi Sukses		End-User akan diteruskan ke halaman login

C. Flow of Event *use case* add ideas

Flow of Event use case add ideas menggambarkan aktivitas yang dilakukan pengguna untuk menambahkan ide. Alur kejadian yang dialamin proses penambahan ide pada saat ide berhasil ditambahkan dan pada saat ide yang tidak *valid* dimasukkan, akan dijelaskan dan diruntutkan pada *Flow of Event* pada Tabel 3.4. Proses kesalahan penambahan ide akan diikuti oleh pemberitahuan jika ide tidak berhasil dimasukkan, yang disebabkan oleh terdapat ide yang sama yang telah dimasukkan sebelumnya dan *input*-an kosong. Ide dimasukkan pada saat pengguna berada di dalam ruangan rapat.

Tabel 3.4 *Flow of Event use case* Add Idea

Usecase Add Idea		
Nama	Add Idea	
Deskripsi Singkat	End-User akan menggunakan Fitur Add Idea untuk menambahkan Ide pada Ruangan Rapat	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User memasuki Ruangan yang telah dimulai oleh moderator
	2	End-User akan mendapatkan informasi ide-ide yang telah ditambahkan sebelumnya
	3	End-User menambahkan ide dan memasukan detail ide
	4	Sistem akan melakukan validasi ide yang telah dimasukan
	5	Jika validasi gagal, maka akan melanjutkan ke Alur Alternatif A1
Alur Alternatif	A1	Aplikasi menampilkan kesalahan inputan ide
Kondisi Sukses		Ide yang telah ditambahkan, akan terlihat di daftar ide-ide

D. Flow of Event *use case* backlog updates

Flow of Event use case add ideas menggambarkan aktivitas yang dilakukan pengguna untuk melihat perubahan dari bacllog. Disini dijelaskan bahwa yang dapat melihat perubahan dari backlog ini hanya moderator. Proses autorisasi apakah pengguna terdaftar di dalam *project* yang sedang dibuka atau tidak juga dijelaskan di dalam *flow of event* pada Tabel 3.5.

Tabel 3.5 *Flow of Event use case* Backlog Updates

Usecase Backlog Updates		
Nama	Backlog Updates	
Deskripsi Singkat	End-User akan menggunakan Fitur Backlog Updates untuk melihat perubahan pada backlog	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	End-User masuk ke dalam halaman project
	2	Aplikasi akan menampilkan daftar Project yang ada
	3	End-User memilih sebuah Project yang diinginkan
	4	Sistem akan melakukan autorisasi untuk End-user atas project
	5	Jika autorisasi tidak berhasil, maka sistem akan melanjutkan pada Alur alternatif A1
	6	End-User akan masuk ke dalam halaman project tersebut
	7	Aplikasi akan memperlihatkan daftar backlog
	8	Memilih Backlog
	9	Detail Backlog dan Progress
Alur Alternatif	A1	Aplikasi akan mengalihkan proses ke alur utama 1
Kondisi Sukses		Pengguna masuk ke dalam aplikasi dan session "current_user" telah terdaftar

E. Flow of Event *use case* chatting

Flow of Event use case chatting menggambarkan aktivitas yang dilakukan pengguna untuk melakukan interaksi *chatting*. Pada saat melakukan *chatting*, seperti yang telah dijelaskan pada Tabel 3.6, akan diberlakukan juga validasi agar data masukan pada proses ini tidak kosong yang akan menyebabkan kesalahan pada aplikasi.

Tabel 3.6 *Flow of Event use case* Chatting

Usecase Chatting		
Nama	Chatting	
Deskripsi Singkat	End-User akan menggunakan Fitur Chatting untuk berinteraksi di dalam ruangan	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User memasuki Ruangan yang telah dimulai oleh moderator
	2	End-User akan mendapatkan informasi interaksi yang telah dilakukan sebelumnya
	3	End-User memasukkan text interaksi
	4	Submit interaksi
Kondisi Sukses		Text yang telah ditambahkan, akan terlihat di LOG chatting

F. Flow of Event *use case* comment ideas

Flow of Event use case comment ideas, menggambarkan aktivitas pengguna di dalam proses menambahkan komentar pada setiap ide yang telah dibuat di dalam ruangan.

Tabel 3.7 *Flow of Event use case* Comment Ideas

Usecase Comment Ideas		
Nama	Comment Ideas	
Deskripsi Singkat	End-User akan menggunakan Fitur Comment Ideas untuk berkomentar atas ide-ide	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	Sistem memeriksa kontrol voting
	2	Jika kontrol voting "enabled", maka akan ke proses nomor 3. Jika kontrol voting "disabled" akan ke proses nomor 4
	3	End-User mengubah status menjadi "disabled"
	4	End-User Mengubah status menjadi "enabled"
Kondisi Sukses		Status kontrol dari voting dapat diubah

G. Flow of Event *use case* enter room

Flow of Event ini menggambarkan aktivitas yang dilakukan peserta rapat untuk memasuki ruang rapat yang telah dimulai oleh moderator. Pada Tabel 3.8 dijelaskan kejadian yang dialami oleh pengguna dalam memasuki ruangan rapat. Bagi peserta rapat, jikalau memasuki ruangan rapat yang belum dimulai oleh moderator, maka seperti alur alternatif A1, akan diarahkan kembali ke halaman *lobby*.

Tabel 3.8 *Flow of Event use case* Control Enter Room

Usecase Enter Room		
Nama	Enter Room	
Deskripsi Singkat	End-User menggunakan fitur ini untuk memasuki ruangan	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam Lobby
	2	Aplikasi akan memperlihatkan daftar ruangan yang ada
	3	End-User memilih ruangan
	4	Jika status ruangan adalah "stoped", maka akan dilanjutkan ke alur alternatif A1
Alur Alternatif	A1	End-User akan diarahkan kembali ke dalam Lobby dengan peringatan bahwa moderator belum mengaktifkan ruangan
Kondisi Sukses		End-User dapat memasuki ruangan

H. Flow of Event *use case* manage user profile

Flow of Event use case logout menggambarkan aktivitas yang terjadi pada proses pengubahan data *profile* dari pengguna. Di dalam proses ini, pengguna dapat melakukan pengubahan terhadap *credential* dari pengguna tersebut seperti, *password, email,* dan *alias.* Pada saat melakukan proses *input* pada aplikasi, validasi juga dilakukan, terutama untuk melakukan pemeriksaan terhadap *password* dan *email*.

Tabel 3.9 *Flow of Event use case* manage user profile

Usecase Manage User Profile		
Nama	Manage User Profile	
Deskripsi Singkat	End-User akan menggunakan Fitur Manage User Profile untuk mengubah *profile* pengguna	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User memasuki halaman *user profile*
	2	End-User akan mendapatkan informasi *profile* tentang dirinya
	3	End-User mengubah *detail profile.*
	4	Sistem akan melakukan validasi ide yang telah dimasukan
	5	Jika validasi gagal, maka akan melanjutkan ke Alur Alternatif A1
Alur Alternatif	A1	Aplikasi menampilkan kesalahan inputan data *profile*
Kondisi Sukses		Aplikasi akan menampilkan notifikasi data berhasil dimasukkan

I. Flow of Event *use case* code snippets

Flow of Event use case code snippets ini menggambarkan aktivitas yang dilakukan pengguna untuk menggunakan fasilitas *snippets* untuk melakukan kolaborasi penyelesaian masalah dalam bentuk *coding*.

Tabel 3.10 *Flow of Event use case* Manage Code Snippets

Usecase Manage Code Snippets		
Nama	Manage Code Snippets	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola *code snippets*	
Aktor	Moderator, Participants	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam halaman code snippets
	2	Aplikasi menampilkan *code snippets* yang telah ada
	3	Jika End-User melakukan penambahan *snippet*, maka akan melakukan alur 4. Tetapi jika melakukan pengelolaan lanjut, akan melakukan alur 6
	4	End-User memasukkan *code snippets*
	5	Jika validasi berhasil, maka aplikasi akan memasukan *code snippets*ke dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	6	Jika End-User melakukan pengubahan snippet, maka akan melakukan alur 7. Jika melakukan penambahan komentar, akan melakukan alur 9
	7	End-User memasukkan data *code snippets*
	8	Jika validasi berhasil, maka aplikasi akan memperbarui snippets, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	9	Memasukkan Komentar pada kolom komentar
	10	Jika validasi berhasil, maka aplikasi akan menambahkan komentar, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan *code snippets*
	2	End-User dapat mengubah *code snippets*
	3	End-User dapat menambahkan komentar

Pada Tabel 3.10,dijelaskan juga kejadian yang terjadi pada saat pengguna pada ruangan rapat menambahkan komentar akan *code snippet* yang telah dimasukkan.

J. Flow of Event *use case* manage impediments

Flow of Event ini menggambarkan aktivitas yang dilakukan pengguna untuk mngelola *impediments*

Gambar 3.11 *Flow of Event use case* Manage Impediments

Usecase Manage impediments		
Nama	Manage impediments	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola impediments	
Aktor	Moderator, Participants	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam halaman impediments
	2	Aplikasi menampilkan impediments yang telah ada
	3	Jika End-User melakukan penambahan, maka akan melakukan alur 4. Tetapi jika melakukan pengelolaan lanjut, akan melakukan alur 6
	4	End-User memasukkan impediments
	5	Jika validasi berhasil, maka aplikasi akan memasukan impedimentske dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	6	Jika End-User melakukan pengubahan, maka akan melakukan alur 7. Jika melakukan penambahan komentar, akan melakukan alur 9
	7	End-User memasukkan data impediments
	8	Jika validasi berhasil, maka aplikasi akan memperbarui impediment, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	9	Memasukkan Komentar pada kolom komentar
	10	Jika validasi berhasil, maka aplikasi akan menambahkan komentar, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan impediments
	2	End-User dapat mengubah impediments
	3	End-User dapat menambahkan komentar

Sama halnya dengan *code snippets*, Tabel 3.11 menjelaskan bahwa fitur *impediments* juga menyediakan fitur penambahan komentar oleh pengguna, yang berguna untuk menyediakan fitur kolaborasi dalam penyelesaian *impediment* pada suatu *project*.

K. Flow of Event *use case* manage participants

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola peserta rapat

Tabel 3.12 *Flow of Event use case* Manage Participant

Usecase Manage Participant		
Nama	Manage Participant	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola peserta rapat	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	End-User masuk ke dalam halaman Project
	2	Aplikasi menampilkan projects yang telah ada
	3	End-User memilih Project yang akan dikelola
	4	Jika End-User ingin menambahkan peserta, maka akan melanjutkan ke langkah 5, tetapi jika ingin mengeluarkan peserta, akan melanjutkan ke langkah 7
	5	End-User menambahkan peserta dengan menggunakan email dari peserta
	6	Sistem akan memeriksa apakah email tersebut tersedia, jika tersedia maka peserta dapat ditambahkan, tetapi jika tidak tersedia akan melakukan langkah A1
	7	End-User melakukan klik pada link "kick"
Alur Alternatif	A1	Aplikasi akan menampilkan peringatan email tidak terdaftar
Kondisi Sukses	1	End-User dapat menambahkan peserta project
	2	End-User dapat mengeluarkan peserta dari project

Pada Tabel 3.12 dijelaskan bahwa moderator dari sebuah *project* mempunyai kemampuan untuk mengelola peserta rapat. Moderator dapat mendaftarkan peserta *project* menggunakan *email.* Tetapi apabila *email* tidak terdaftar, maka akan diberikan pemberitahuan jika *email* dari peserta yang dimasukkan, tidak terdaftar. Selain itu, moderator juga dapat mengeluarkan peserta dari *project.*

L. Flow of Event *use case* manage project

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola *project*

Tabel 3.13 *Flow of Event use case* Manage Project

Usecase Manage projects		
Nama	Manage projects	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola projects	
Aktor	Moderator, Participants	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam halaman projects
	2	Aplikasi menampilkan projects yang telah ada
	3	Jika End-User melakukan penambahan projects, maka akan melakukan alur 4. Tetapi jika melakukan pengubahan, akan melakukan alur 7
	4	End-User memasukkan projects
	5	Jika validasi berhasil, maka aplikasi akan memasukan projects ke dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	6	Aplikasi akan menambahkan *sprint* pertama untuk project yang telah terbuat
	7	End-User memasukkan data projects
	8	Jika validasi berhasil, maka aplikasi akan memperbarui snippets, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan projects
	2	End-User dapat mengubah projects

M. Flow of Event *use case* manage room

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola ruangan

Tabel 3.14 *Flow of Event use case* Manage Room

Usecase Manage rooms		
Nama	Manage rooms	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola rooms	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai project moderator	
Alur Utama	1	End-User masuk ke lobby
	2	Aplikasi menampilkan rooms yang telah ada
	3	Jika End-User melakukan penambahan rooms, maka akan melakukan alur 4. Tetapi jika melakukan perubahan status rooms, akan melakukan alur 6
	4	End-User memasukkan rooms
	5	Jika validasi berhasil, maka aplikasi akan memasukan rooms ke dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	6	End-User mengubah status ruangan ("enable" / "disable")
	7	Aplikasi mengubah status rooms
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan rooms
	2	End-User dapat mengubah status rooms

N. Flow of Event *use case* manage sprints

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk mengelola *sprint*. Serta juga menggambarkan aktivitas yang dilakukan oleh peserta rapat untuk me-*review sprint*.

Tabel 3.15 *Flow of Event use case* Manage Sprints

Usecase Manage sprints		
Nama	Manage sprints	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola sprint	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai project moderator	
Alur Utama	1	End-User masuk ke halaman projects
	2	Aplikasi menampilkan projects yang telah ada
	3	End-User memilih Project yang diinginkan
	4	Aplikasi menampilkan sprint yang terdaftar di dalam project tersebut
	5	Jika End-User melakukan penambahan sprint, maka akan melakukan alur 6. Tetapi jika melakukan perubahan status sprint, akan melakukan alur 6
	6	End-User memasukkan sprint
	7	Jika validasi berhasil, maka aplikasi akan memasukan sprint ke dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
	8	End-User mengubah status ruangan ("enable" / "disable")
	9	Aplikasi mengubah status sprint
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan sprint
	2	End-User dapat mengubah status sprint

O. Flow of Event *use case* progress reporting

Flow of Event ini menggambarkan aktivitas yang dilakukan pengguna dalam melakukan pelaporan perkembangan pengembangan perangkat lunak pada setiap *sprint*. Kegiatan yang ditunjukkan pada Tabel 3.16 ini menjelaskan proses pelaporan yang akan dilakukan oleh pengguna pada saat melakukan rapat. Status dari sebuah tugas akan diubah, beserta memberikan penjelasan tentang perkembangan pengerjaan tugas yang dilakukan sampai pada rapat itu dilaksanakan.

Tabel 3.16 *Flow of Event use case* Progress Reporting

Usecase Progress Reporting		
Nama	Progress Reporting	
Deskripsi Singkat	End-User menggunakan fitur ini untuk melakukan pelaporan perkembangan tugas	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke lobby
	2	Aplikasi menampilkan rooms yang telah ada
	3	End-User memasuki ruangan yang dituju
	4	Aplikasi akan menampilkan data dari tugas yang telah dimasukkan sebelumnya
	5	End-User memilih tugas yang ingin dilaporkan
	6	End-User mengubah status perkembangan tugas tersebut
	7	Aplikasi mengubah status dari tugas tersebut
Kondisi Sukses	1	End-User dapat mengubah status dari perkembangan tugas

P. Flow of Event *use case* run sprint

*Flow o*f Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk menjalankan *sprint*. Dengan menjalankan *sprint*, maka rapat dari *sprint* dapat dilakukan.

Tabel 3.17 *Flow of Event use case* Run Sprint

Usecase Run Sprint		
Nama	Run Sprint	
Deskripsi Singkat	End-User menggunakan fitur ini untuk menjalankan sprint	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	End-User masuk ke dalam halaman projects
	2	Aplikasi menampilkan projects yang telah ada
	3	End-User memilih project
	3	Aplikasi menampilkan daftar sprint pada project tersebut
	4	End-User menjalankan sprint
	5	Jika validasi berhasil, maka aplikasi akan menjalankan sprint, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
Alur Alternatif	A1	Aplikasi akan menampilkan bahwa masih terdapat sprint yang masih berjalan
Kondisi Sukses	1	End-User dapat menjalankan sprint

Q. Flow of Event *use case* sprint reviews

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk melakukan *review* terhadap *sprint*, history aktivitas dan penyelesaian di dalam *sprint* juga ditampilkan.

Tabel 3.18 *Flow of Event use case* Sprint Reviews

Usecase Sprint Review		
Nama	Sprint Review	
Deskripsi Singkat	End-User menggunakan fitur ini untuk melihat *history*dari sebuah sprint	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam halaman projects
	2	Aplikasi menampilkan projects yang telah ada
	3	End-User memilih project
	3	Aplikasi menampilkan daftar sprint pada project tersebut
	4	End-User melihat *history* dari sebuah sprint
	5	Jika sprint masih berjalan, maka aplikasi akan mengarah ke alur A1, tetapi jika tidak, aplikasi akan menampilkan *history*dari sprint
Alur Alternatif	A1	Aplikasi menampilkan peringatan bahwa sprint masih berjalan
Kondisi Sukses	1	End-User dapat melihat *history* sprint

R. Flow of Event *use case* start meeting

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk memulai rapat. Dalam memulai rapat, *project moderator* dapat hanya melakukan klik pada link "*run meeting*". Pada saat moderator menjalankan rapat pada suatu ruangan, peserta rapat akan dapat memasuki ruangan rapat tersebut.

Tabel 3.19 *Flow of Event use case* Start Meeting

Usecase Start Meeting		
Nama	Start Meeting	
Deskripsi Singkat	End-User menggunakan fitur ini untuk memulai rapat di dalam ruangan	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	End-User masuk ke dalam Lobby
	2	Aplikasi akan memperlihatkan daftar ruangan yang ada
	3	End-User memilih ruangan
	4	End-User akan memulai rapat dengan keterangan "Sprint Planing" atau "Daily Meeting"
Kondisi Sukses	1	Moderator dapat memulai rapat
	2	Participant dapat memasuki ruangan

S. Flow of Event *use case* stop meeting

Flow of Event ini menggambarkan aktivitas yang dilakukan *project moderator* untuk menghentikan rapat.

Tabel 3.20 *Flow of Event use case* Stop Meeting

Usecase Stop Meeting		
Nama	Stop Meeting	
Deskripsi Singkat	End-User menggunakan fitur ini untuk menghentikan rapat di dalam ruangan	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, sebagai moderator	
Alur Utama	1	End-User masuk ke dalam Lobby
	2	Aplikasi akan memperlihatkan daftar ruangan yang ada
	3	End-User memilih ruangan
	4	End-User akan menghentikan rapat dalam suatu ruangan
Kondisi Sukses	1	Moderator dapat menghentikan rapat
	2	Participant tidak dapat memasuki ruangan

T. Flow of Event *use case* voting on ideas

Flow of Event ini menggambarkan aktivitas yang dilakukan pengguna dalam melakukan voting untuk setiap ide yang ada. *Voting* yang d

Tabel 3.21 *Flow of Event use case* Voting on ideas

Usecase Voting on Ideas		
Nama	Voting on Ideas	
Deskripsi Singkat	End-User menggunakan fitur ini untuk melakukan voting	
Aktor	Moderator, Participant	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke lobby
	2	Aplikasi menampilkan rooms yang telah ada
	3	End-User memasuki ruangan yang dituju
	4	Aplikasi akan menampilkan data dari ide-ide yang telah dimasukkan sebelumnya
	5	End-User memilih ide untuk melakukan voting
	6	Aplikasi akan menampilkan detail dari ide yang dipilih, beserta komentar-komentar
	7	End-User memilih "like" atau "dislike"
Kondisi Sukses	1	End-User dapat melakukan voting

Voting dilakukan pada setiap ide yang diajukan oleh peserta rapat. *Voting* akan berupa status "*like*" atau "*dislike*". *Voting* terdapat pada *sprint planning*. Hasil dari voting dapat digunakan dalam melakukan sortir ide pada saat melakukan finalisasi *backlog* pada *sprint meeting*.

U. Flow of Event *use case* backlog finalization

Flow of Event ini menggambarkan aktivitas yang dilakukan moderator untuk melakukan seleksi ide-ide untuk dijadikan *backlogs*.

Tabel 3.22 *Flow of Event use case* backlog finalization

Usecase Backlog Finalization		
Nama	Backlog Finalization	
Deskripsi Singkat	Moderator menggunakan fitur ini untuk melakukan seleksi ide-ide untuk dijadikan ***backlogs***	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, dan sebagai moderator	
Alur Utama	1	Moderator masuk ke ruangan rapat
	2	Melakukan klik pada link "Collect Ideas"
	3	Aplikasi akan menampilkan data dari ide-ide yang telah dimasukkan dan di vote sebelumnya
	4	Moderator kemudian melakukan pemilihan pada ide yang diinginkan, maka akan ke proses nomor 5. Tetapi jika ingin membatalkan pilihan akan ke proses nomor 6
	5	Aplikasi akan mengubah status ide tersebut menjadi *backlog candidate*
	6	Aplikasi akan mengubah status ide tersebut menjadi nil
	7	Setelah selesai, moderator melakukan submit *backlogs*
	8	Aplikasi akan membuat *backlogs*
Kondisi Sukses	1	Moderator dapat memilih ide
	2	Moderator dapat membatalkan ide yang terpilih
	3	*Project Backlogs* dapat terbuat

V. Flow of Event *use case* sprint task finalization

Flow of Event ini menggambarkan aktivitas yang dilakukan moderator untuk melakukan seleksi *backlogs* untuk dijadikan *sprint tasks*

Tabel 3.23 *Flow of Event use case* sprint task finalization

Usecase Sprint Task Finalization		
Nama	Sprint Task Finalization	
Deskripsi Singkat	Moderator menggunakan fitur ini untuk melakukan seleksi backlog-backlog untuk dijadikan sprint tasks	
Aktor	Moderator	
Prasyarat	End-User telah melakukan login terlebih dahulu, dan sebagai moderator	
Alur Utama	1	Moderator masuk ke ruangan rapat
	2	Melakukan klik pada link "Collect Backlogs"
	3	Aplikasi akan menampilkan data dari backlog-backlog yang telah dimasukkan dan di vote sebelumnya
	4	Moderator kemudian melakukan pemilihan pada backlog yang diinginkan, maka akan ke proses nomor 5. Tetapi jika ingin membatalkan pilihan akan ke proses nomor 6
	5	Aplikasi akan mengubah status backlog tersebut menjadi sprint task candidate
	6	Aplikasi akan mengubah status backlog tersebut menjadi nil
	7	Setelah selesai, moderator melakukan submit sprint tasks
	8	Aplikasi akan membuat sprint tasks
Kondisi Sukses	1	Moderator dapat memilih backlog
	2	Moderator dapat membatalkan backlog yang terpilih
	3	Project Sprint Tasks dapat terbuat

.

W. Flow of Event *use case* manage file sharing

Flow of Event ini menggambarkan aktivitas yang dilakukan pengguna dalam mengelola dan berbagi *files* dalam satu ruangan.

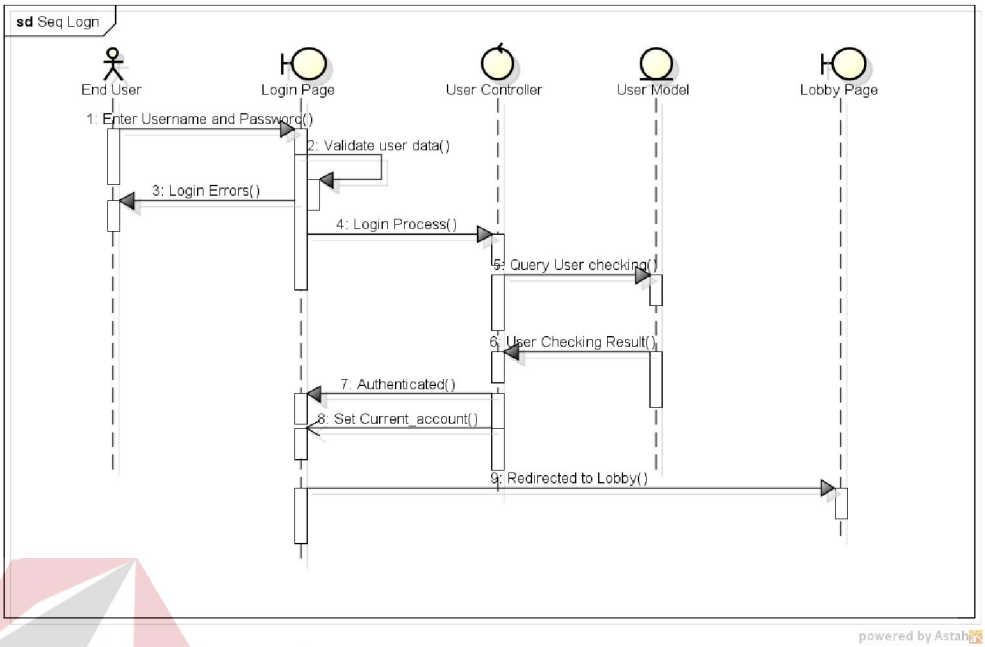
Tabel 3.24 *Flow of Event use case* manage file sharing

Usecase Manage file sharing		
Nama	Manage file sharing	
Deskripsi Singkat	End-User menggunakan fitur ini untuk mengelola files	
Aktor	Moderator, Participants	
Prasyarat	End-User telah melakukan login terlebih dahulu	
Alur Utama	1	End-User masuk ke dalam halaman files
	2	Aplikasi menampilkan files yang telah ada
	3	End-User memasukkan files
	4	Jika validasi berhasil, maka aplikasi akan memasukan files ke dalam database, tetapi jika tidak berhasil, maka akan mengarah ke alur A1
Alur Alternatif	A1	Aplikasi akan menampilkan kesalahan pemasukan data
Kondisi Sukses	1	End-User dapat menambahkan files
	2	End-User dapat mengubah files

3.3.6 *Sequence Diagram*

A. Sequence Diagram *use case* login

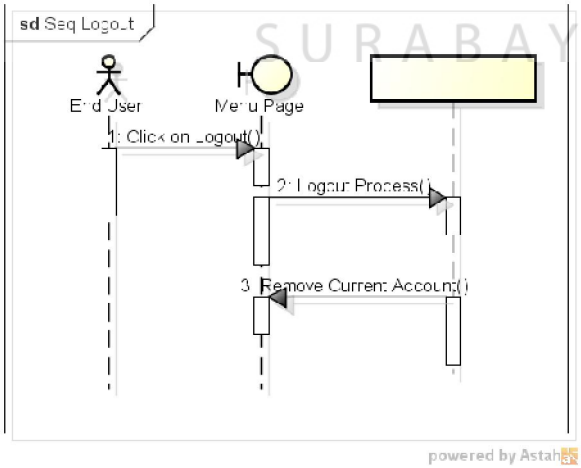
Pada proses ini, pengguna memasukkan *username* dan *password* pada halaman *login*. Setelah itu *User controller* akan melakukan klarifikasi data kepada *User model*. Jika klarifikasi yang dilakuka telah berhasil dan pengguna telah terdaftar di dalam aplikasi, maka *User controller* akan melakukan *set session "current_account"*.



Gambar 3.33 *Sequence diagram use case login*

B. *Sequence Diagram use case logout*

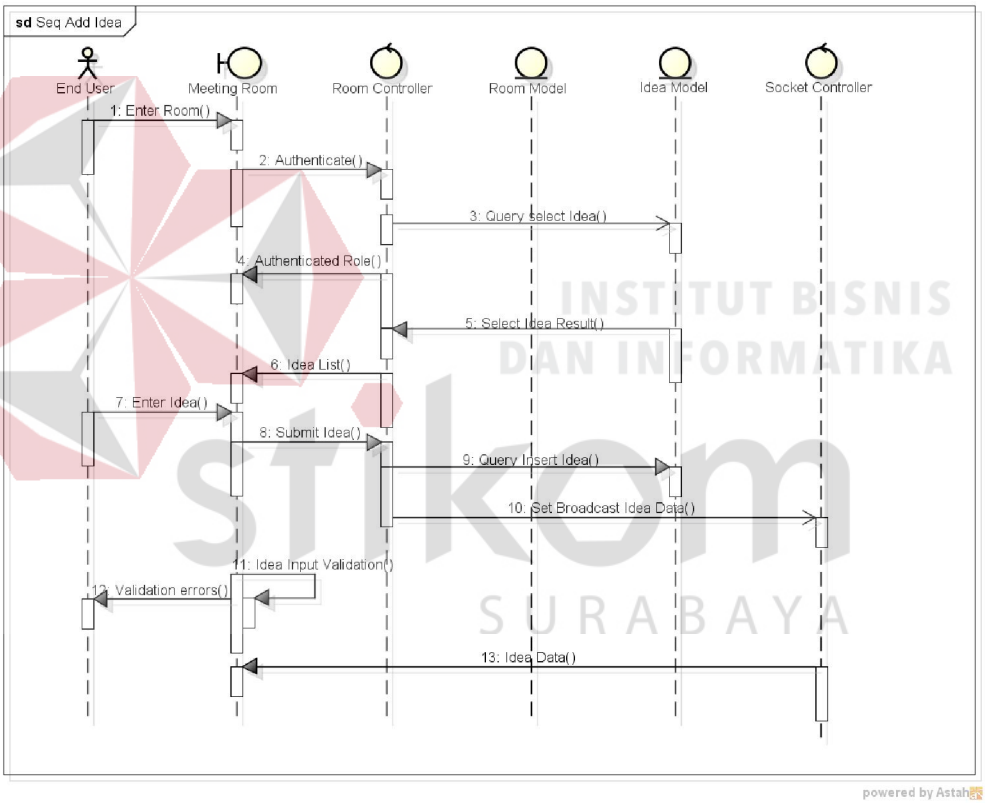
Pada proses ini, pengguna melakukan klik pada *link logout* pada halaman menu. Jika proses berhasil, maka aplikasi akan menghapus *session "current_account"*



Gambar 3.34 *Sequence diagram use case logout*

C. *Sequence Diagram use case add idea*

Pada proses ini, pengguna dapat menambahkan ide pada ruangan rapat dimana pengguna berada. Ide dimasukkan melalui *Room Controller*, lalu aplikasi akan melakukan *Insert* pada *Model Idea*. Jika proses ini berhasil, maka data dari ide yang telah dimasukkan tersebut akan dikirimkan ke dalam *socket controller* untuk selanjutnya disampaikan (*broadcast)* kepada pengguna yang lain pada ruangan rapat yang sama.

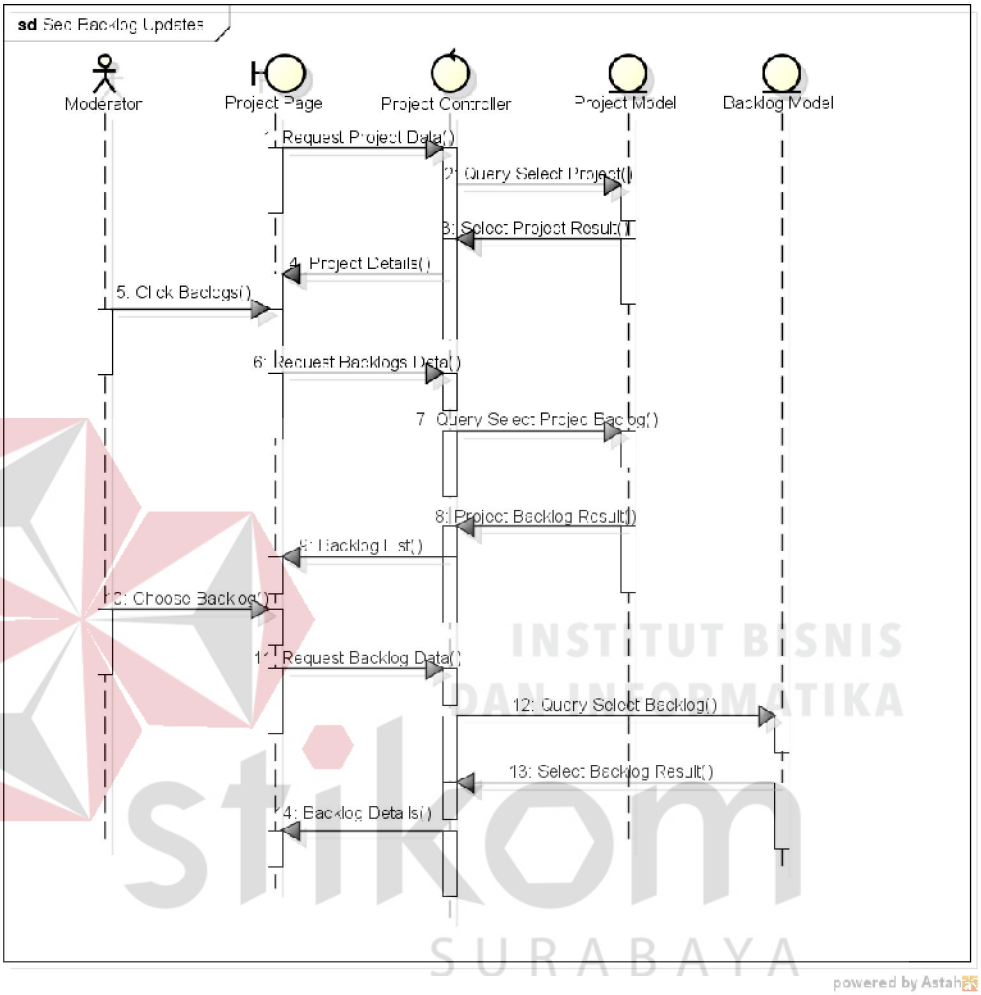


Gambar 3.35 *Sequence diagram use case add idea*

D. *Sequence Diagram use case backlog updates*

Proses ini merupakan proses yang dilakukan oleh *project moderator* untuk melakukan pengawasan pekerjaan pada suatu *project*. Pada halaman *project*, akan

ditampilkan daftar dari *backlog*. Moderator dapat memilih salah satu yang diinginkan untuk melihat detail dari *backlog* tersebut.

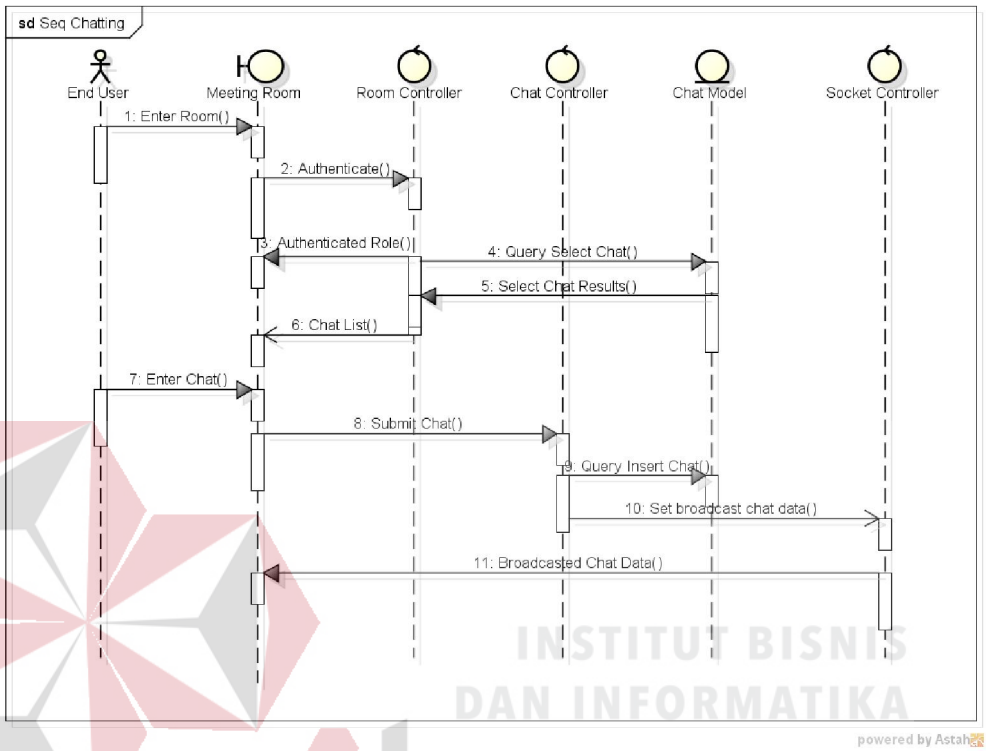


Gambar 3.36 *Sequence diagram use case backlog updates*

E. *Sequence Diagram use case chatting*

Proses ini merupakan proses interaksi berupa pengiriman pesan secara *realtime* pada ruangan rapat (*chat)*. Pengguna memasukkan pesan yang akan disampaikan, pada *box chat* yang tersedia pada ruangan rapat, lalu diproses oleh *Chat Controller*, baru setelah itu disampaikan pada *chat model*. Setelah proses penyimpanan *chat* berhasil, maka proses akan diteruskan oleh *Socket*

controller untuk melakukan *broadcast* kepada pengguna lain pada ruangan rapat yang sama.

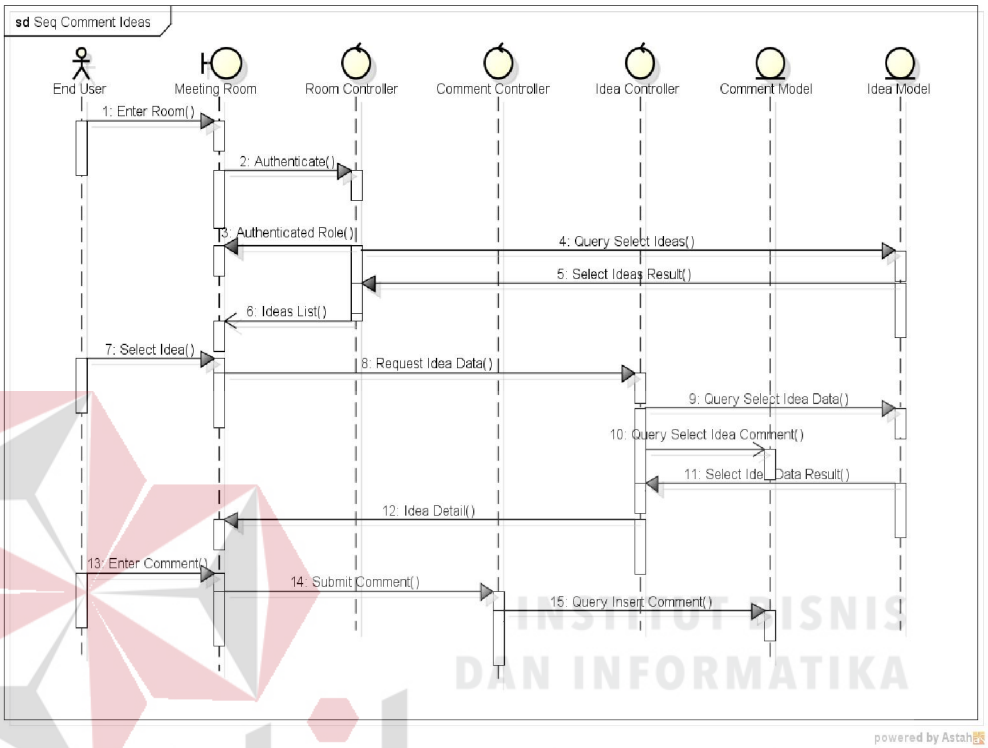


Gambar 3.37 *Sequence diagram use case chatting*

F. *Sequence Diagram use case comment ideas*

Proses ini merupakan proses dimana pengguna, di dalam ruangan rapat yang sama, dapat memberikan komentar pada setiap ide yang telah diajukan oleh peserta rapat. Pengguna memilih salah satu ide yang telah ditampilkan oleh aplikasi. Aplikasi akan meminta *Idea controller* untuk memberikan *response* terhadap permintaan tersebut. *Idea controller* kemudian melakukan permintaan kepada *Idea model* untuk berhubungan dengan *database*. Setelah menerima hasil dari pencarian data, kemudian aplikasi akan menampilkan detil dari ide yang dipilih. Pengguna dapat membaca detil dari ide tersebut, beserta dengan komentar-komentar yang telah diajukan untuk ide tersebut.

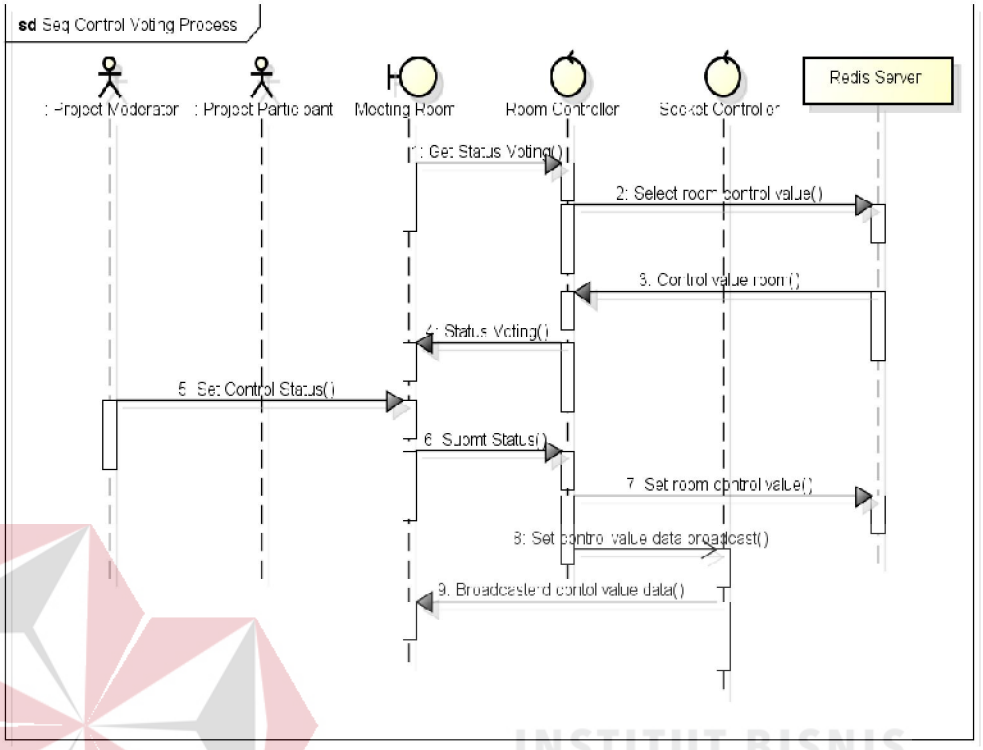
Pengguna dapat memasukkan komentar pada *box* inputan yang telah disediakan.



Gambar 3.38 *Sequence diagram use case comment ideas*

G. *Sequence Diagram use case control voting and proposing ideas*

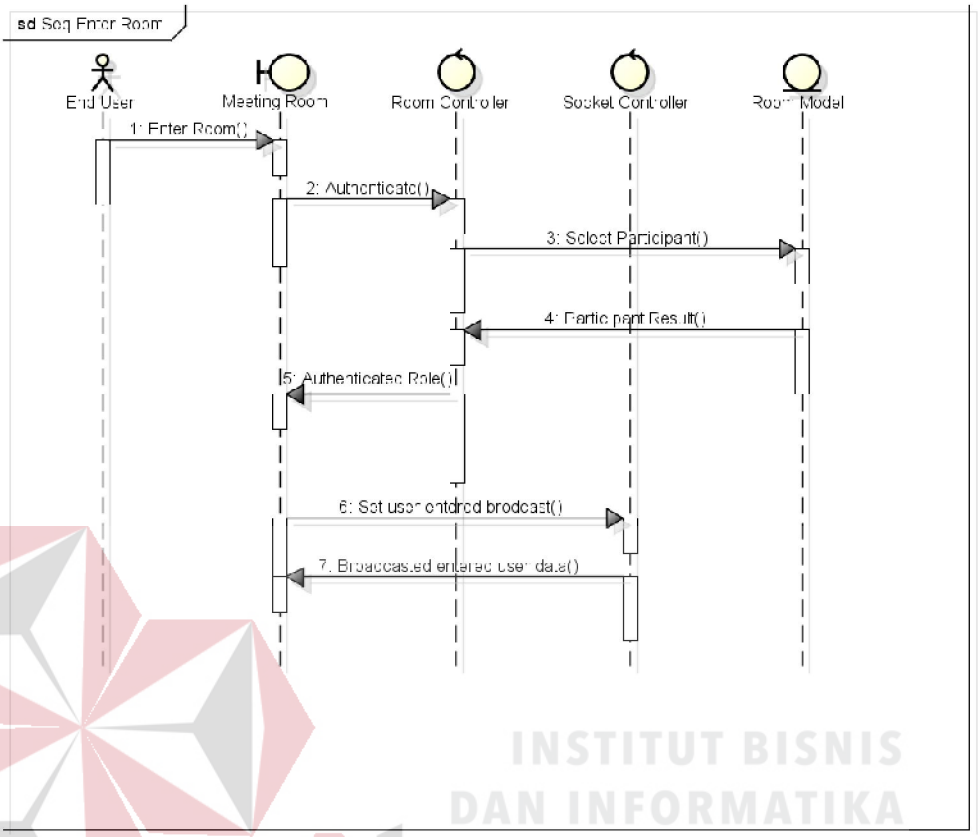
Proses ini merupakan proses moderasi yang akan dilakukan oleh *project* moderator untuk mengendalikan jalannya rapat pada suatu ruangan. Moderator akan mengeset status interaksi yang ada pada ruangan rapat. Pada saat pengesetan berhasil, maka *Socket controller* akan meneruskan pemberitahuan perubahan status interaksi.



Gambar 3.39 *Sequence diagram use case control and proposing ideas*

H. *Sequence Diagram use case enter room*

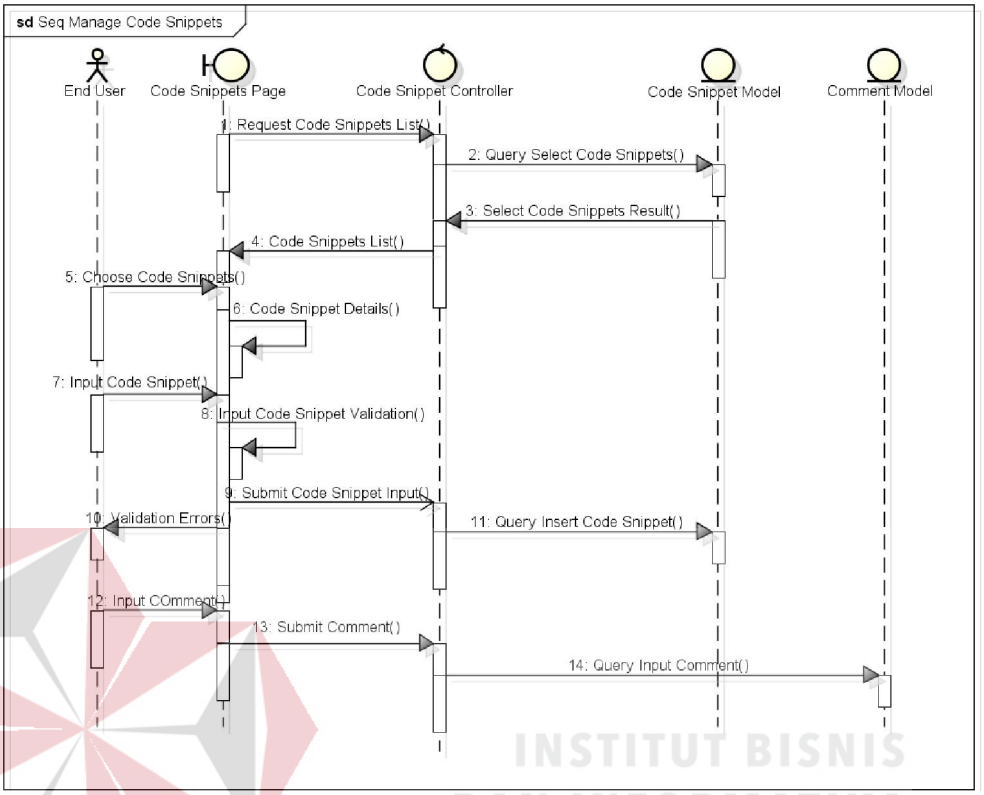
Merupakan proses memasuki ruangan. Pada proses ini, *room controller* melakukan autentikasi terhadap pengguna yang bergabung ke dalam ruangan rapat. Autentikasi yang dilakukan adalah melakukan cek apakah pengguna tersebut terdaftar di dalam *project* pada ruangan rapat tersebut ataukah tidak. Pada waktu pengguna berhasil memasuki ruangan, maka peserta yang ada di dalam ruangan tersebut akan mendapatkan informasi bahwa terdapat peserta yang baru saja bergabung secara *real time* yang akan dilakukan oleh *socket controller*.



Gambar 3.40 *Sequence diagram use case enter room*

I. *Sequence Diagram use case manage code snippets*

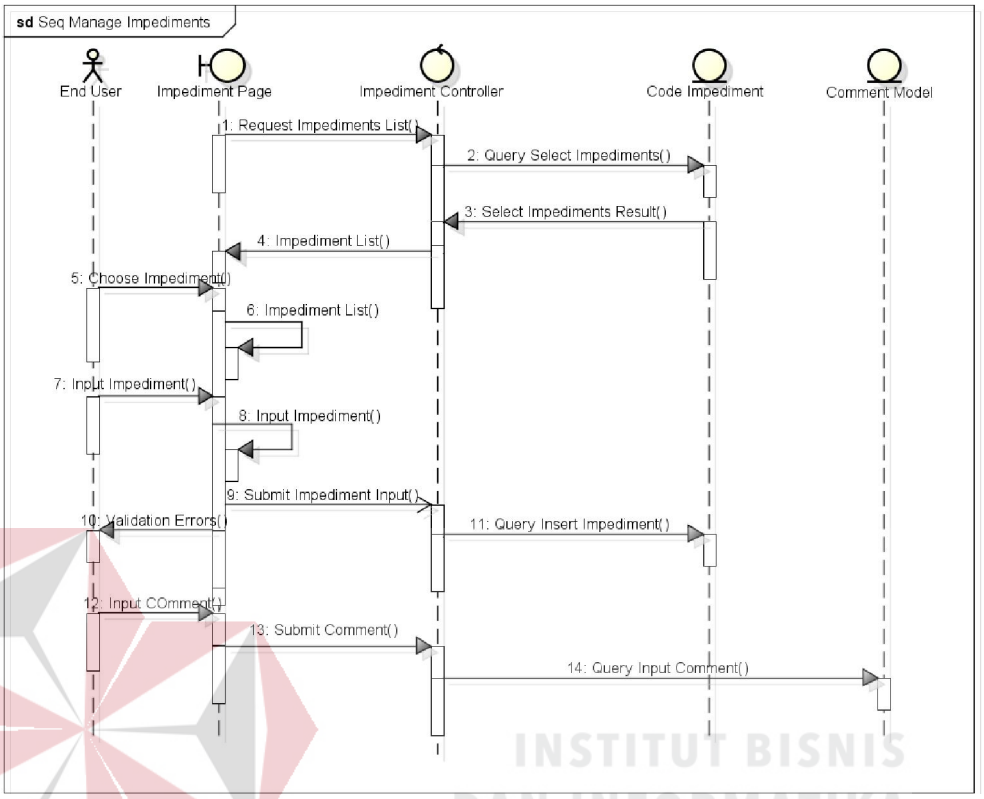
Merupakan proses untuk mengelola *code snippets*. Terdapat halaman tersendiri untuk mengelola *code snippets*. Pengguna dapat memasukkan *code snippets*. Pada waktu pengguna berhasil memasukan data dari *code snippet*, maka peserta yang ada di dalam ruangan tersebut akan mendapatkan informasi bahwa terdapat *code snippet* yang baru saja ditambahkan secara *real time.* Fitur pemberian komentar pada *code snippet* juga dijelaskan pada Gambar 3.41.



Gambar 3.41 *Sequence diagram use case code snippets*

J. *Sequence Diagram use case manage impediments*

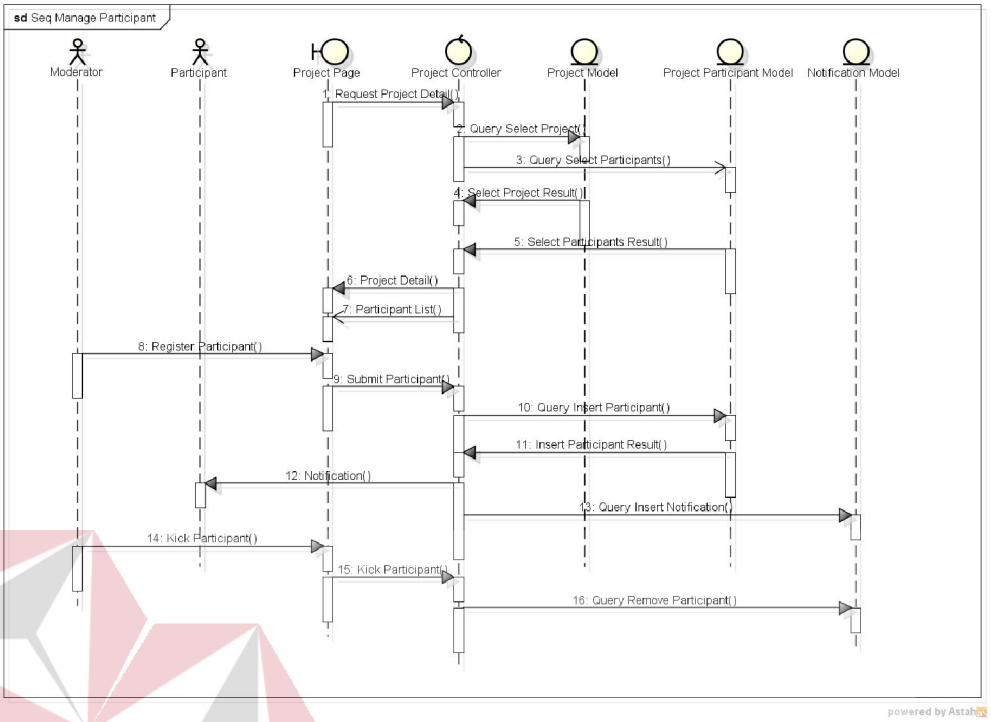
Merupakan proses untuk mengelola kesulitan (*impediment)* pada project . Pengguna dapat menambahkan *impediment* dengan disertai komentar-komentar yang dapat dipergunakan sebagai diskusi penyelesaian dari *impediment* tersebut. Pada waktu pengguna berhasil memasukan data dari *impediment*, maka peserta yang ada di dalam ruangan tersebut akan mendapatkan informasi bahwa terdapat *impediment* yang baru saja ditambahkan secara *real time.* Fitur pemberian komentar pada *impediment* juga dijelaskan pada Gambar 3.42.



Gambar 3.42 *Sequence diagram use case manage impediments*

K. *Sequence Diagram use case manage participant*

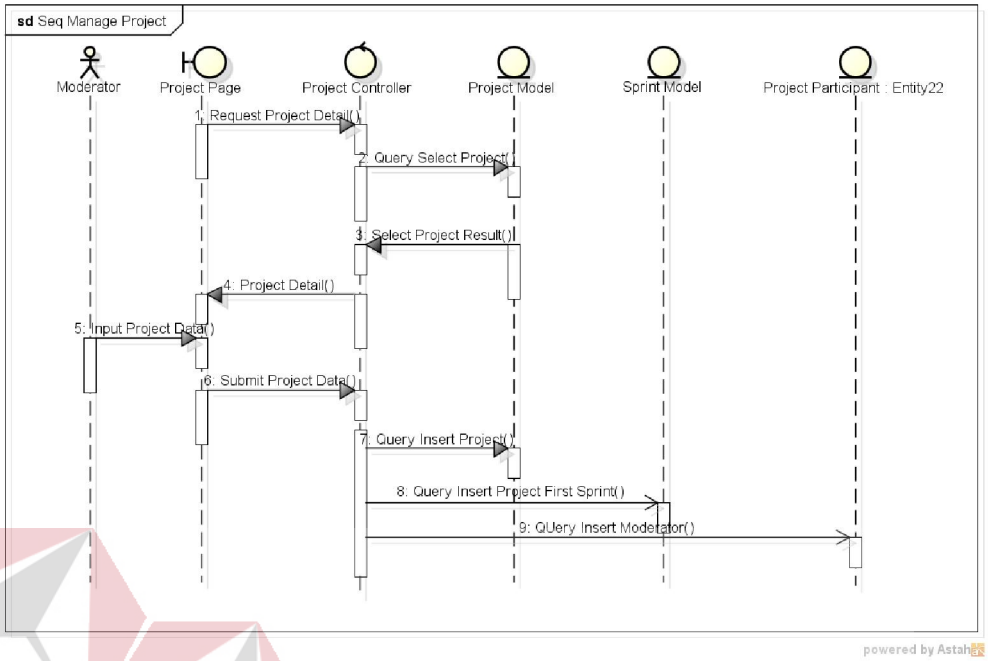
Moderator dapat melakukan pengelolaan peserta dari setiap project. Moderator dapat melakukan registrasi peserta ke dalam suatu *project*. Pada saat melakukan registrasi peserta, setelah proses regirstrasi berhasil, maka peserta yang baru saja di regirstrasi akan mendapatkan notifikasi dengaan informasi telah teregistrasi pada suatu *project*. Selain itu moderator juga dapat mengeluarkan peserta dari *project.*



Gambar 3.43 *Sequence diagram use case manage participants*

L. *Sequence Diagram use case manage project*

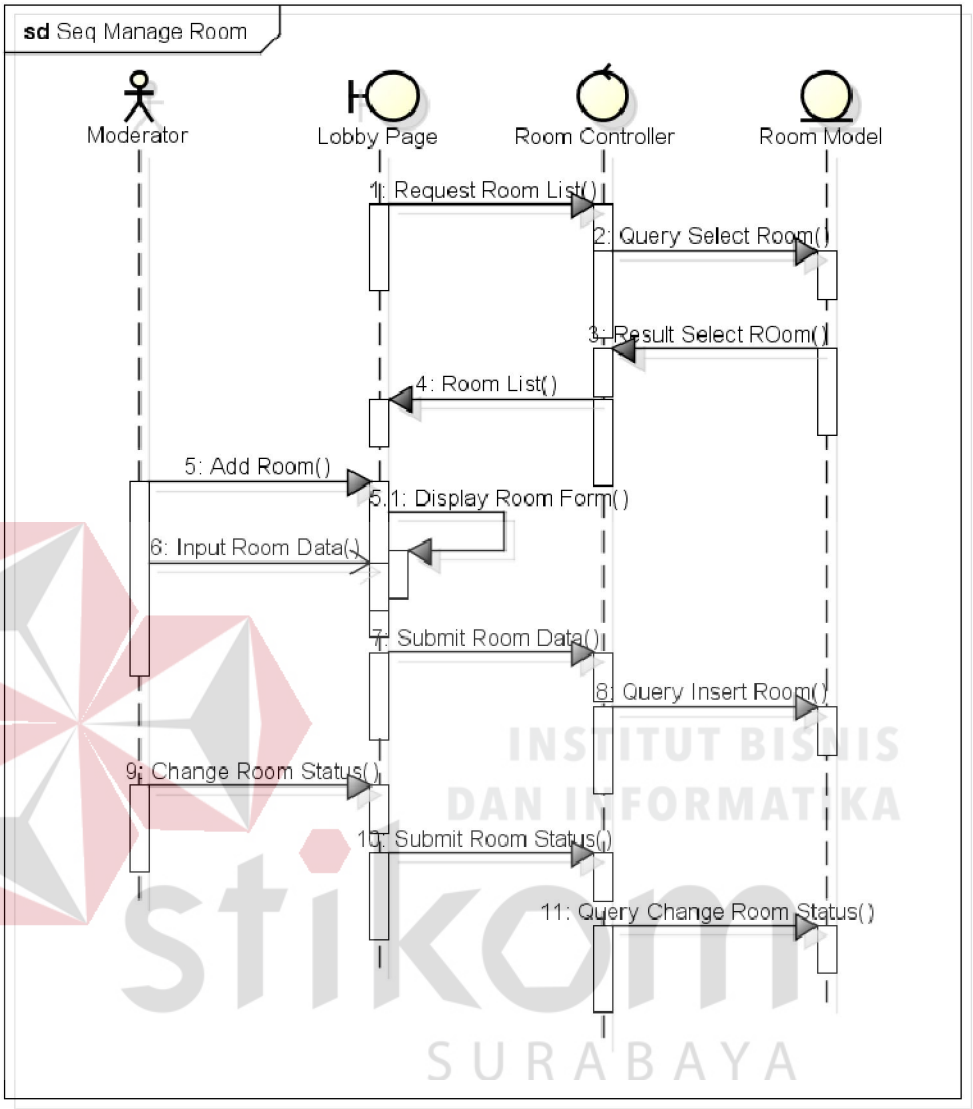
Pengguna dapat melakukan pengelolaan pada *project*. Pada saat pengguna membuat suatu *project* baru, pengguna tersebut akan secara langsung menjadi moderator dari *project* tersebut. *Sprint* pertama dari *project* yang baru saja terbuat akan secara langsung dibuat di dalam database. Pada Gambar 3.43, dijelaskan bahwa dalam melakukan pengelolaan sebuah *project,* moderator dapat memasukkan peserta *projet* dengan menggunakan *email*. Pada waktu melakukan penambahan peserta, *project controller* kemudian melakukan cek apakah peserta yang dimasukkan telah terdaftar atau belum. Setelah berhasil menambahkan peserta, maka aplikasi akan mengirimkan notifikasi kepada pengguna bahwa pengguna tersebut telah ditambahkan ke dalam *project* tersebut.



Gambar 3.44 *Sequence diagram use case manage projects*

M. *Sequence Diagram use case manage room*

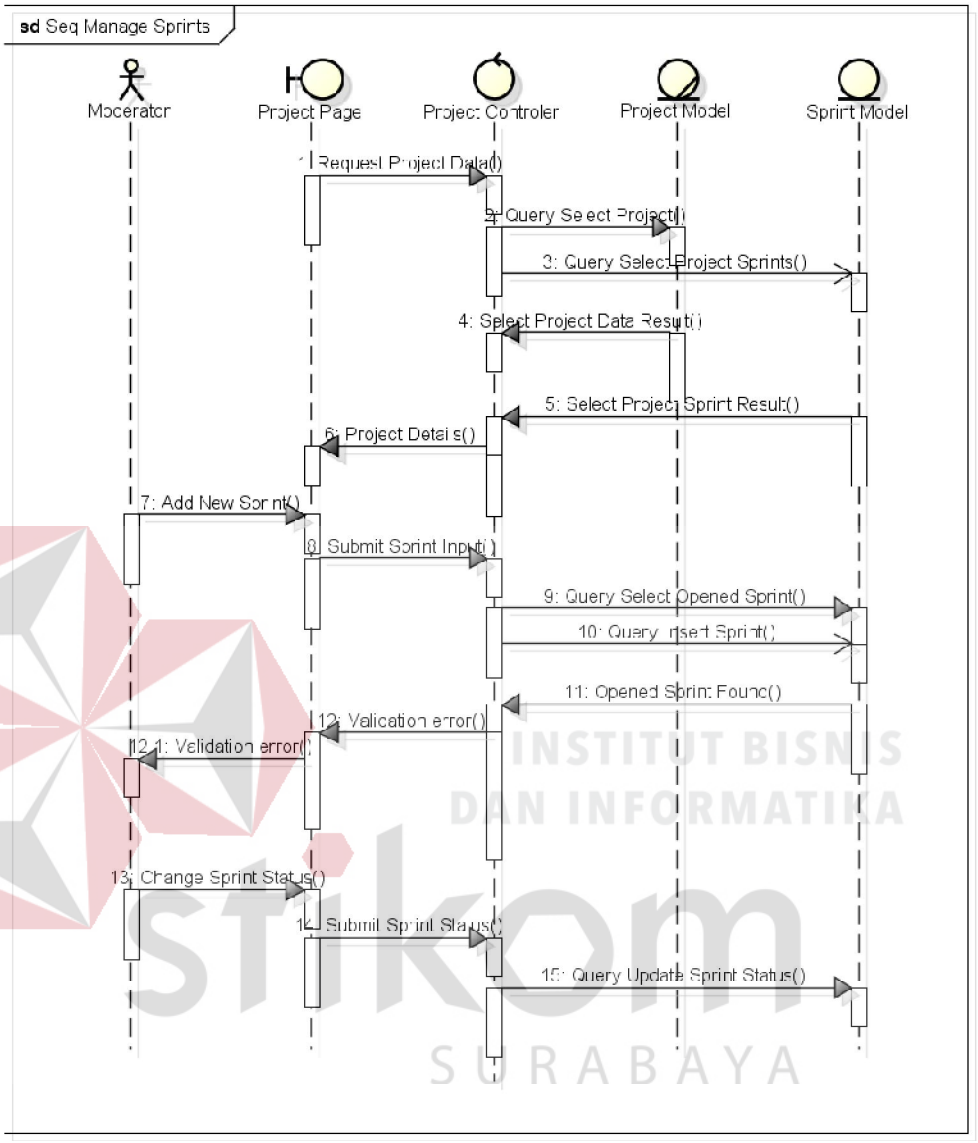
Pada saat akan memulai rapat untuk mendiskusikan *sprint*, moderator harus membuat ruangan rapat dahulu berdasarkan *sprint* yang akan didiskusikan. Jika akan memulai rapat harian, moderator dapat menyalakan ruangan untuk dipakai rapat. Dengan begitu peserta rapat dapat memasuki ruangan rapat. Pada saat ruangan rapat dijalankan oleh moderator, aplikasi akan mengirimkan data *update* kepada *room model* berupa status yang dapat digunakan untuk melakukan *update* terhadap status ruangan rapat tersebut.



Gambar 3.45 *Sequence diagram use case manage room*

N. *Sequence Diagram use case manage sprint*

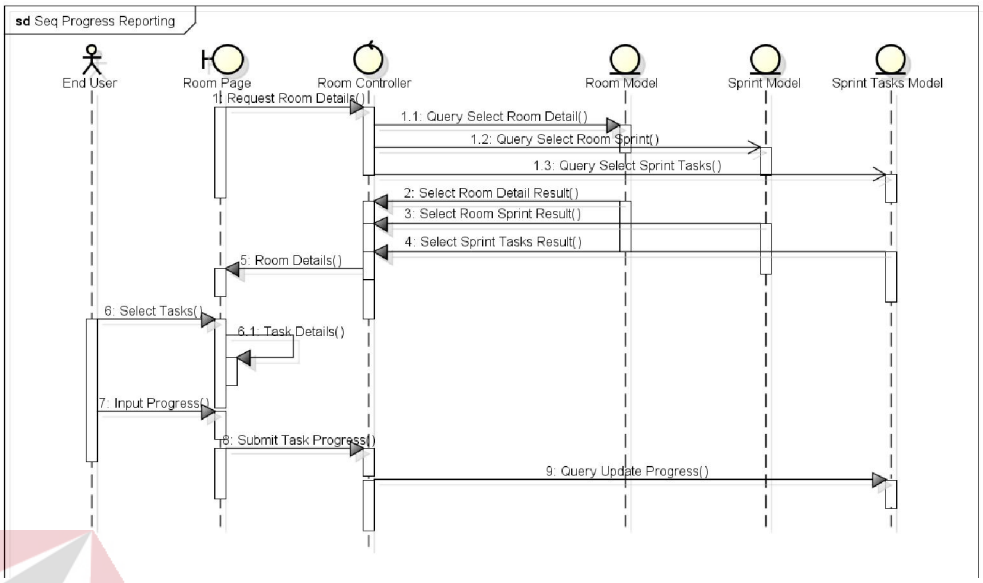
Di dalam *project*, jika *sprint* telah selesai pada batas waktunya, maka *sprint* tersebut dapat ditutup untuk menyatakan bahwa *sprint* tersebut telah berakhir. Selanjutnya, *sprint* baru akan dibuat untuk menyelesaikan *project* tersebut. Tetapi jika masih ada *sprint* yang masih berjalan, maka *sprint* yang baru tidak dapat dibuat.



Gambar 3.46 *Sequence diagram use case manage sprint*

O. *Sequence Diagram use case progress reporting*

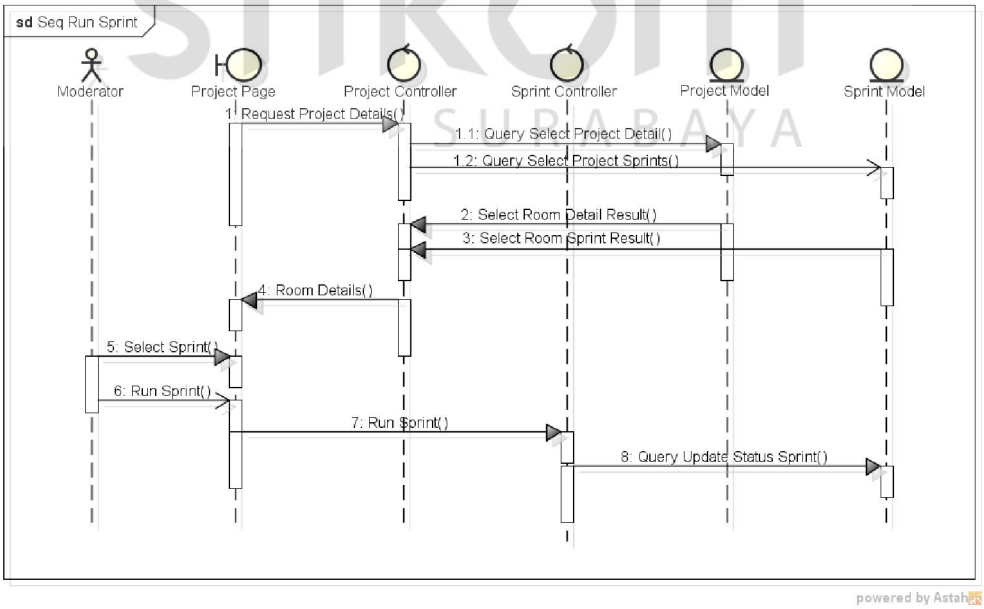
Pada saat melakukan pekerjaan masing-masing, terdapat proses yang dapat diangkakan dalam bentuk prosentase. Maka dari itu, pada saat rapat harian prosentase perkembangan dari pekerjaan yang telah diselesaikan pada hari sebelumnya harus dilaporkan untuk mengetahui perkembangan dari setiap pekerjaan.



Gambar 3.47 *Sequence diagram use case progress reporting*

P. *Sequence Diagram use case run sprint*

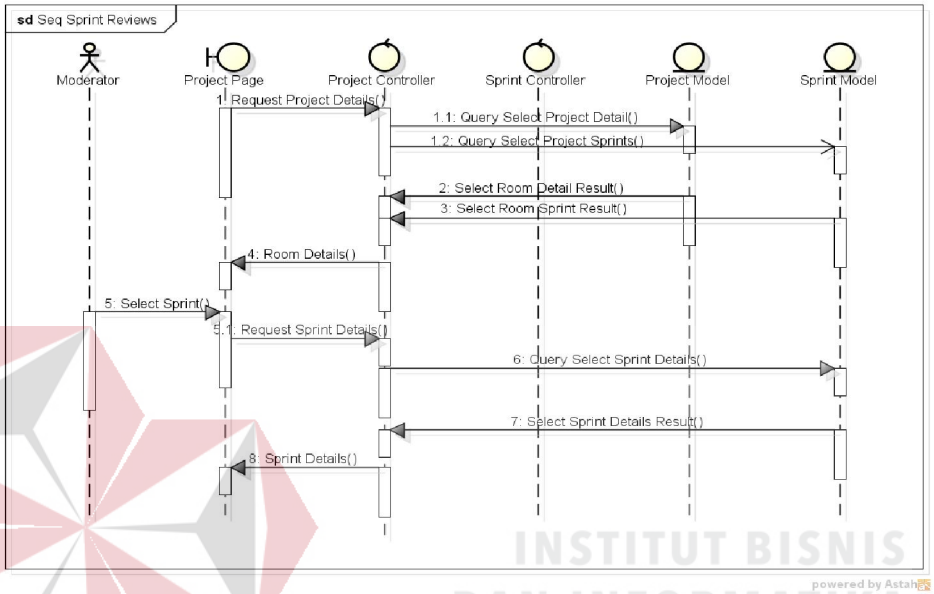
Proses ini digunakan moderator untuk menjalankan *sprint* yang baru saja terbuat atau *sprint* lama yang telah ditutup sebelumnya.



Gambar 3.48 *Sequence diagram use case run sprint*

Q. *Sequence Diagram use case sprint review*

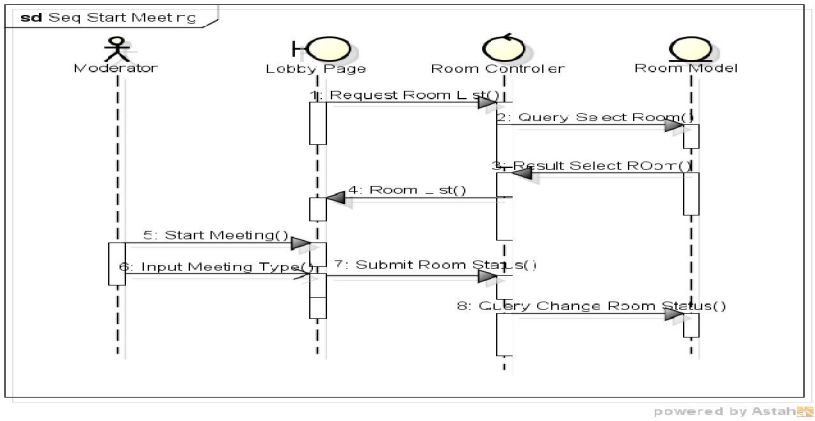
Proses ini digunakan pengguna untuk dapat melakukan *view* atas *sprint* yang telah / sedang berjalan.



Gambar 3.49 *Sequence diagram use case sprint review*

R. *Sequence Diagram use case start meeting*

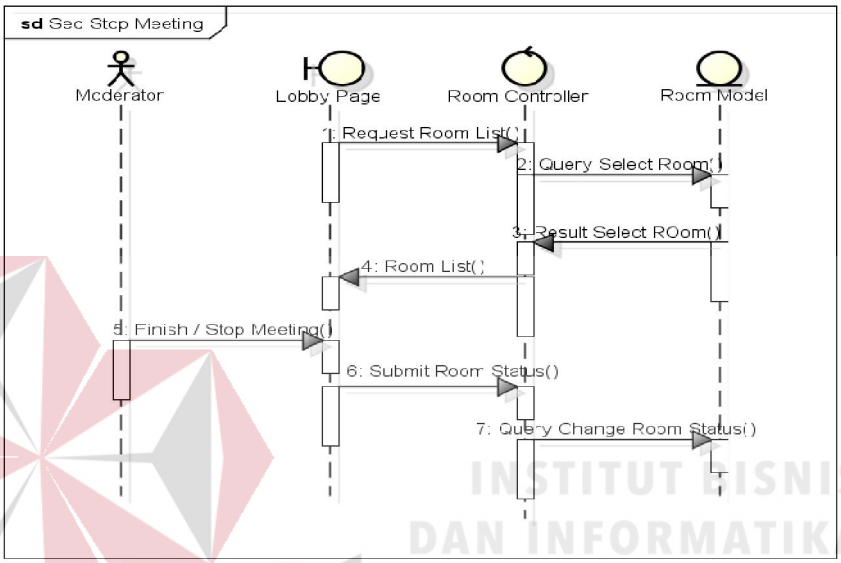
Proses ini dilakukan moderator pada ruangan rapat yang ada untuk menyalakan ruangan yang kemudian akan dipakai untuk melaksanakan rapat.



Gambar 3.50 *Sequence diagram use case start meeting*

S. *Sequence Diagram use case stop meeting*

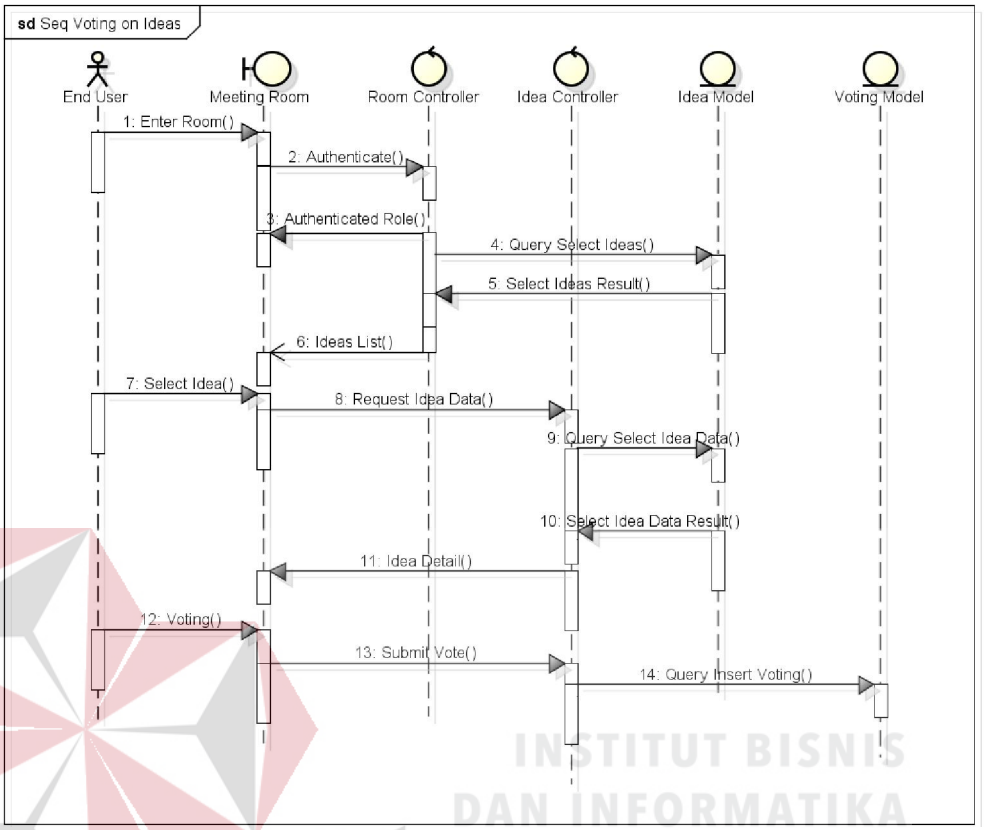
Proses ini digunakan moderator untuk menonaktifkan ruangan setelah dipakai untuk mengadakan rapat. Dengan begitu, peserta tidak dapat memasuki ruangan tersebut.



Gambar 3.51 *Sequence diagram use case stop meeting*

T. *Sequence Diagram use case voting on ideas*

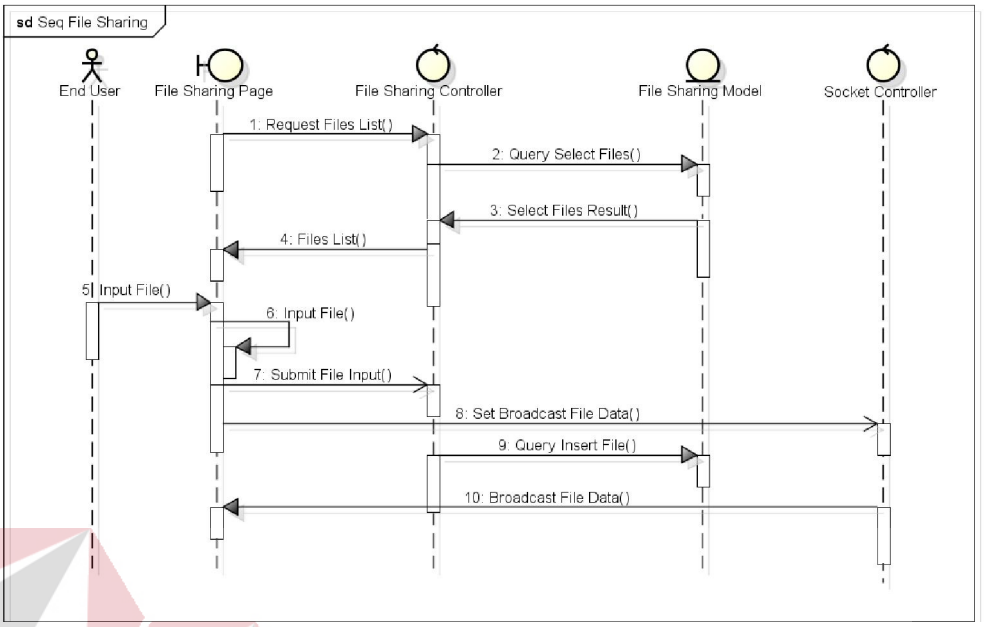
Di dalam ruangan rapat, pengguna dapat melakukan voting atas ide yang telah diajukan. Voting ini berupa *like* dan *dislike*, yang nantinya akan dipergunakan moderator untuk melakukan sortir dalam finalisasi ide. Sebelum melakukan vote, aplikasi akan menampilkan detil dari ide yang akan devoting, sehingga pengguna dapat mengetahui secara detil maksud dan penjelasan dari ide tersebut.



Gambar 3.52 *Sequence diagram use case vote on ideas*

U. *Sequence Diagram use case file sharing*

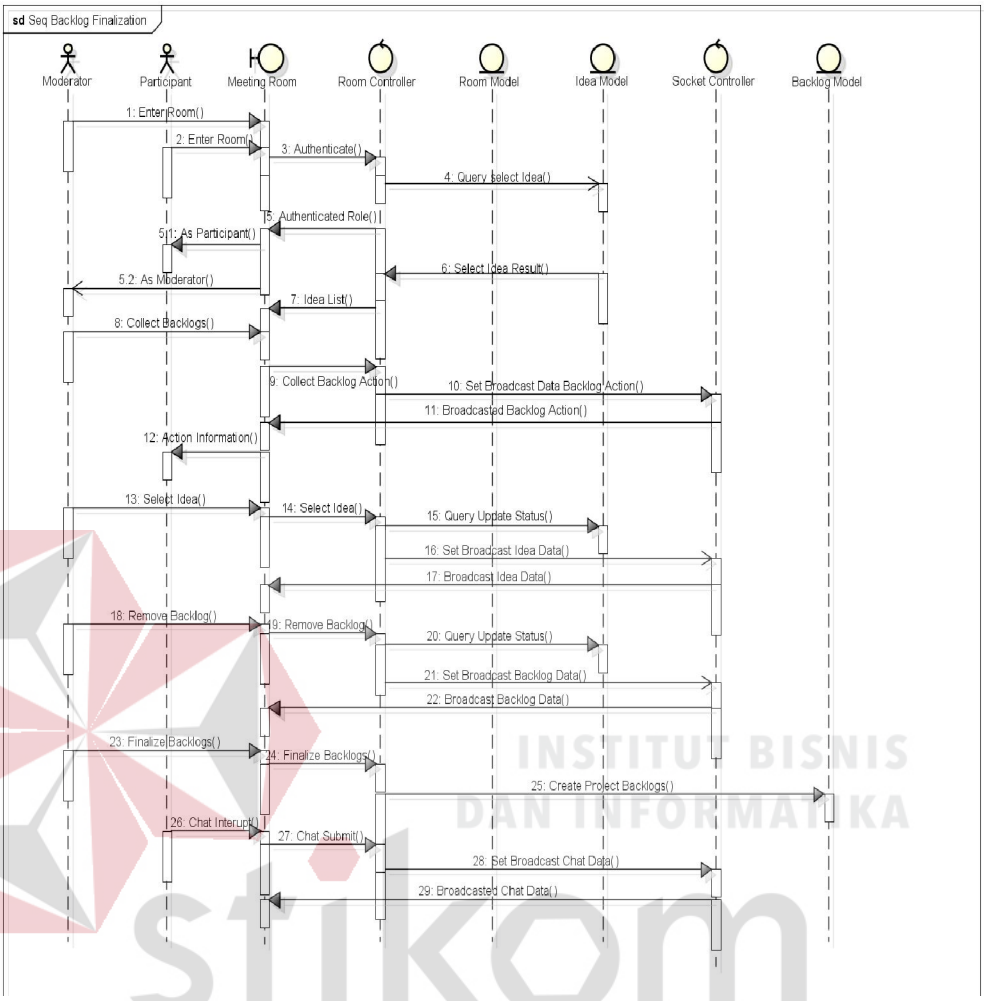
Di dalam ruangan rapat, pengguna dapat melakukan *sharing file* satu sama lainnya. Pengguna dapat melakukan *upload file* pada bagian *file sharing* Pada waktu pengguna berhasil melakukan *upload*, maka peserta yang ada di dalam ruangan tersebut akan mendapatkan informasi bahwa terdapat *file* yang baru saja ditambahkan. Selain itu, peserta dapat mengunduh *file* yang telah ter*upload,* seperti dijelaskan pada Gambar 3.53.



Gambar 3.53 *Sequence diagram use case manage file sharing*

V. *Sequence Diagram use case backlog finalization*

Proses ini merupakan pemilihan ide yang telah divote dan telah diajukan oleh pengguna menjadi *Project Backlogs*. Moderator yang bertugas untuk melakukan pemilihan, dengan dibantu oleh fasilitas sortir berdasarkan jumlah voting pada setiap ide. Pada waktu moderator melakukan pemilihan ide oleh moderator secara *realtime*, setiap peserta dari rapat tidak dapat melakukan hal lain selain melihat ide yang telah dipilih oleh moderator dan juga melakukan *chat*, yang memungkinkan peserta melakukan interupsi pada saat pemilihan ide tengah berlangsung. Setelah ide dipilih, maka finalisasi *baklog* dapat dijalankan dan *Project Backlog* telah terbuat.

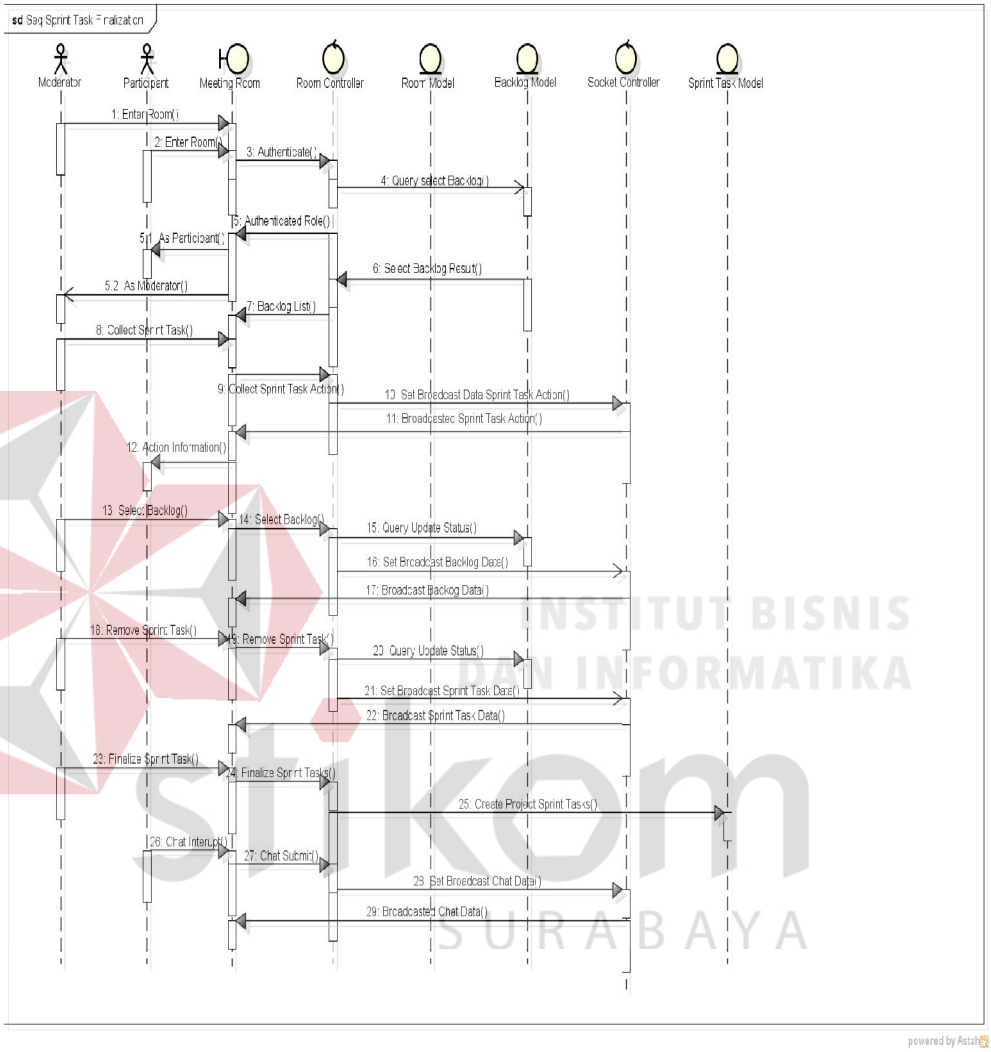


Gambar 3.54 *Sequence diagram use case backlog finalization*

W. *Sequence Diagram use case sprint task finalization*

Proses ini merupakan pemilihan backlog yang telah di*vote* dan telah diajukan oleh pengguna menjadi Sprint Task. Moderator yang bertugas untuk melakukan pemilihan, dengan dibantu oleh fasilitas sortir berdasarkan jumlah voting pada setiap *backlog*. Pada waktu moderator melakukan pemilihan *backlog*, setiap peserta dari rapat tidak dapat melakukan hal lain selain melihat proses pemilihan *backlog* oleh moderator secara *realtime* dan juga melakukan *chat*, yang memungkinkan peserta melakukan interupsi pada saat pemilihan

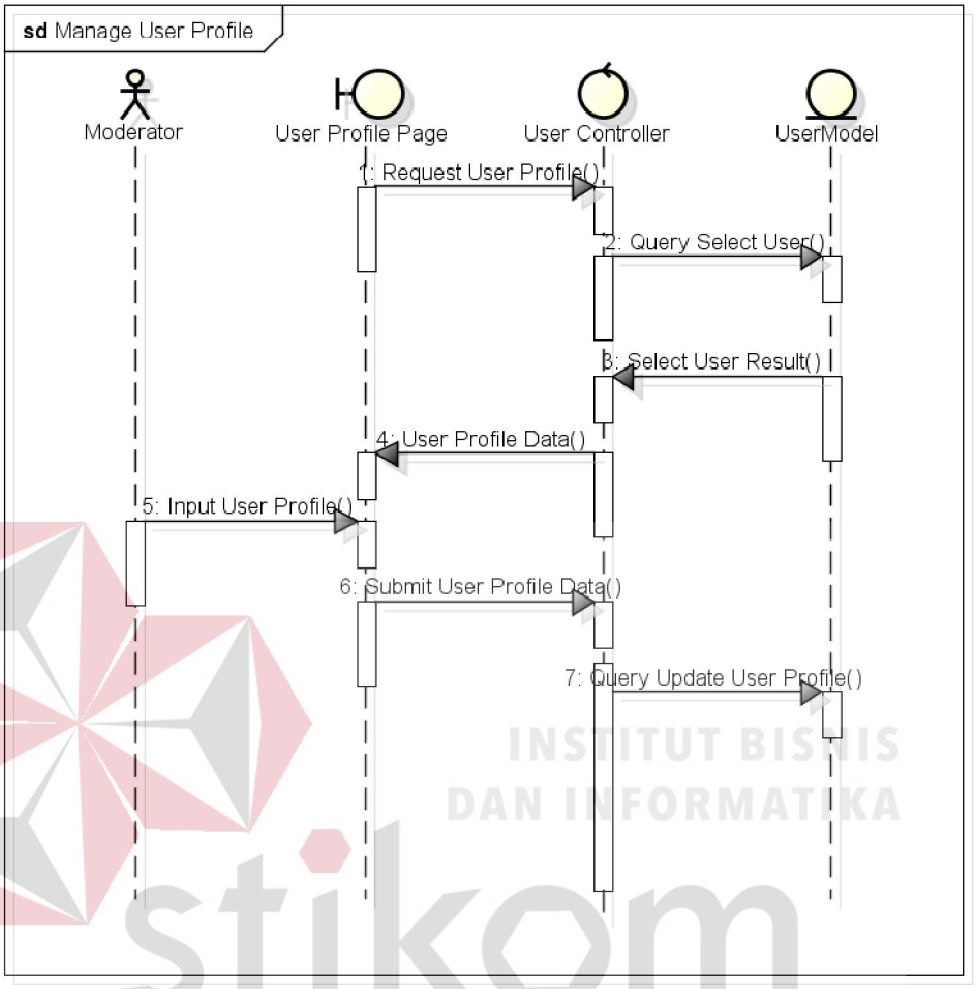
ide tengah berlangsung. Setelah *backlog* telah selesai dipilih, maka finalisasi *sprint task* dapat dijalankan dan *Project Sprint Task* telah terbuat.



Gambar 3.55 *Sequence diagram use case sprint task finalization*

X. *Sequence Diagram use case user profile*

Proses ini merupakan fasilitas bagi pengguna untuk dapat merubah data penting berkaitan dengan pengguna tersebut, seperti password dan email.



Gambar 3.56 *Sequence diagram use case manage user profile*

3.3.7 Menemukan diagram kelas

Sebelum membuat diagram kelas penulis melakukan pendaftaran objek yang akan menjadi kelas dengan cara memperhatikan *flow of event* dan diagram sekuensial. Berikut merupakan kandidat kelas yang telah diamati dari diagram sekuensial.

Tabel 3.25 kandidat kelas pada tiap diagram sekuensial

Flow of Event	Kandidat Kelas	Jenis
Add Idea	Meeting Room	Boundary
	Room Controller	Entity
	Room Model	Entity
	Idea Model	Entity
	Socket Controller	Entity
Backlog Finalization	Meeting Room	Boundary
	Room Controller	Entity
	Room Model	Entity
	Idea Model	Entity
	Socket Controller	Entity
	Backlog Model	Entity
Backlog Updates	Project Page	Boundary
	Project Controller	Entity
	Project Model	Entity
	Backlog Model	Entity
Chatting	Meeting Room	Boundary
	Room Controller	Entity
	Chat Controller	Entity
	Chat Model	Entity
	Socket Controller	Entity
Comment Ideas	Meeting Room	Boundary
	Room Controller	Entity
	Comment Controller	Entity
	Idea Controller	Entity
	Comment Model	Entity
	Idea Model	Entity
Control Voting Process	Meeting Room	Boundary
	Room Controller	Entity
	Socket Controller	Entity
	Redis Server	Entity
Enter Room	Meeting Room	Boundary
	Room Controller	Entity
	Socket Controller	Entity
	Room Model	Entity
Login	Login Page	Boundary
	User Controller	Entity
	User Model	Entity
	Lobby Page	Entity

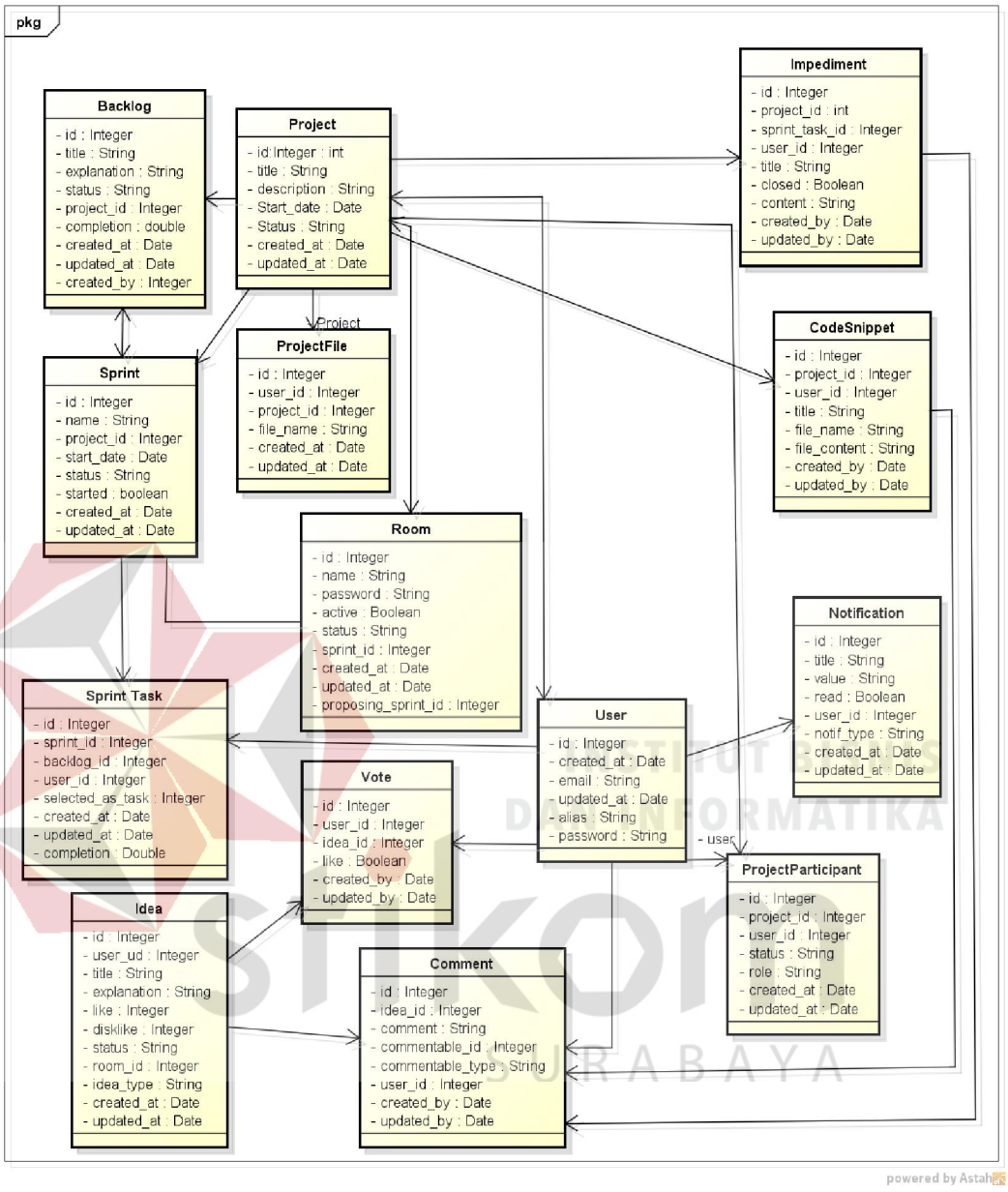
Flow of Event	Kandidat Kelas	Jenis
Logout	Menu Page	Boundary
	User Controller	Entity
Manage Code Snippets	Code Snippets Page	Boundary
	Code Snippets Controller	Entity
	Code Snippet Model	Entity
	Comment Model	Entity
Manage File Sharing	File Sharing Page	Boundary
	File Sharing Controller	Entity
	File Sharing Model	Entity
	Socket Controller	Entity
Manage Impediments	Impediment Page	Boundary
	Impediment Controller	Entity
	Impediment Model	Entity
	Comment Model	Entity
Manage Participant	Project Page	Entity
	Project Model	Entity
	Projec Participant Model	Entity
	Notification Model	Entity
Manage Project	Project Page	Boundary
	Project Controller	Entity
	Project Model	Entity
	Sprint Model	Entity
	Project Participant	Entity
Manage Room	Lobby Page	Boundary
	Room Controller	Entity
	Room Model	Entity
Manage Sprints	Project Page	Boundary
	Project Controller	Entity
	Project Model	Entity
	Sprint Model	Entity
Progress Reporting	Room Page	Boundary
	Room Controller	Entity
	Room Model	Entity
	Sprint Model	Entity
	Sprint Tasks Model	Entity

Flow of Event	Kandidat Kelas	Jenis
Run Sprint	Project Page	Boundary
	Project Controller	Entity
	Sprint Controller	Entity
	Project Model	Entity
	Sprint Model	Entity
Sprint Reviews	Project Page	Boundary
	Project Controller	Entity
	Sprint Controller	Entity
	Project Model	Entity
	Sprint Model	Entity
Sprint Task Finalization	Meeting Room	Boundary
	Room Controller	Entity
	Room Model	Entity
	Backlog Model	Entity
	Socket Controller	Entity
	Sprint Tasks Model	Entity
Start Meeting	Lobby Page	Boundary
	Room Controller	Entity
	Room Model	Entity
Stop Meeting	Lobby Page	Boundary
	Room Controller	Entity
	Room Model	Entity
Voting on Ideas	Meeting Room	Boundary
	Room Controller	Entity
	Idea Controller	Entity
	Idea Model	Entity
	Voting Model	Entity

3.3.8 *Class diagram*

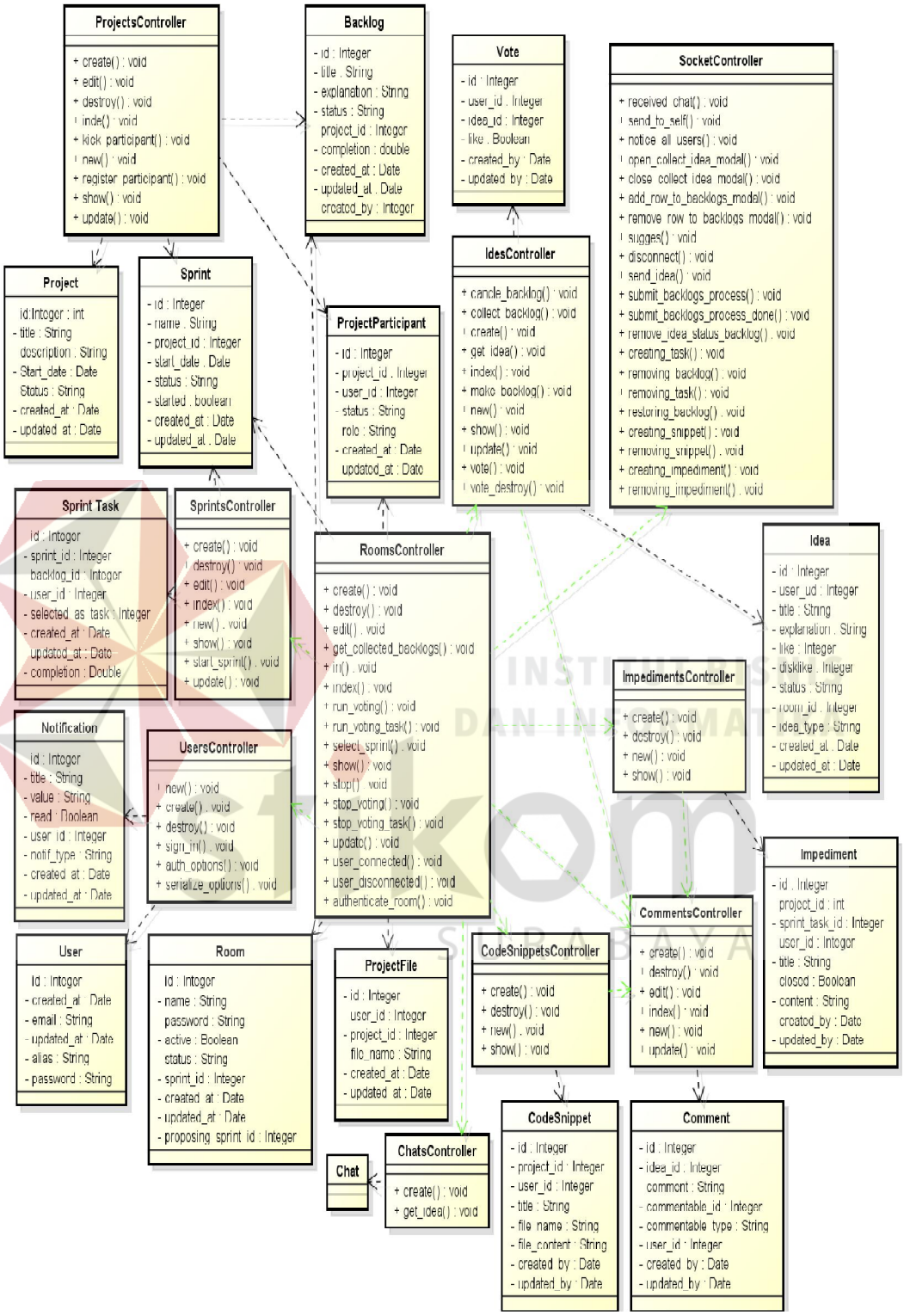
Pembuatan *class diagram* yang diperlukan dalam perancangan aplikasi rapat *online*, akan dibahas dalam sub-bab ini. Diagram kelas akan dipisah menjadi 3 bagian, dikarenakan banyaknya *class* serta *boundary* yang digunakan di dalam perancangan aplikasi. Bagian pertama menggambarkan relasi antar kelas *model* yang merupakan *Object Data* di dalam aplikasi.. Bagian kedua menggambarkan hubungan antara *Model-Controller* di dalam aplikasi, dimana *model* akan memberikan *parsing* data kepada *controller* berdasarkan *request* yang disampaikan. Bagian ketiga menggambarkan hubungan *View-Controller* yang digunakan oleh sistem untuk melakukan permintaan dari *client* ke *controller*.

Pada perancangan diagram kelas terdapat 3 kelas utama yaitu *model*, *view*, *controller*. Pada *Ruby on Rails*, *model* adalah kelas yang berhubungan langsung dengan database dan *model* juga mewakili *Object Data*. Pada *class diagram* hubungan antar model akan digambarkan relasi antar satu *model* dengan *model* lainnya. *Class diagram* hubungan antar model dapat dilihat pada Gambar 3.57.



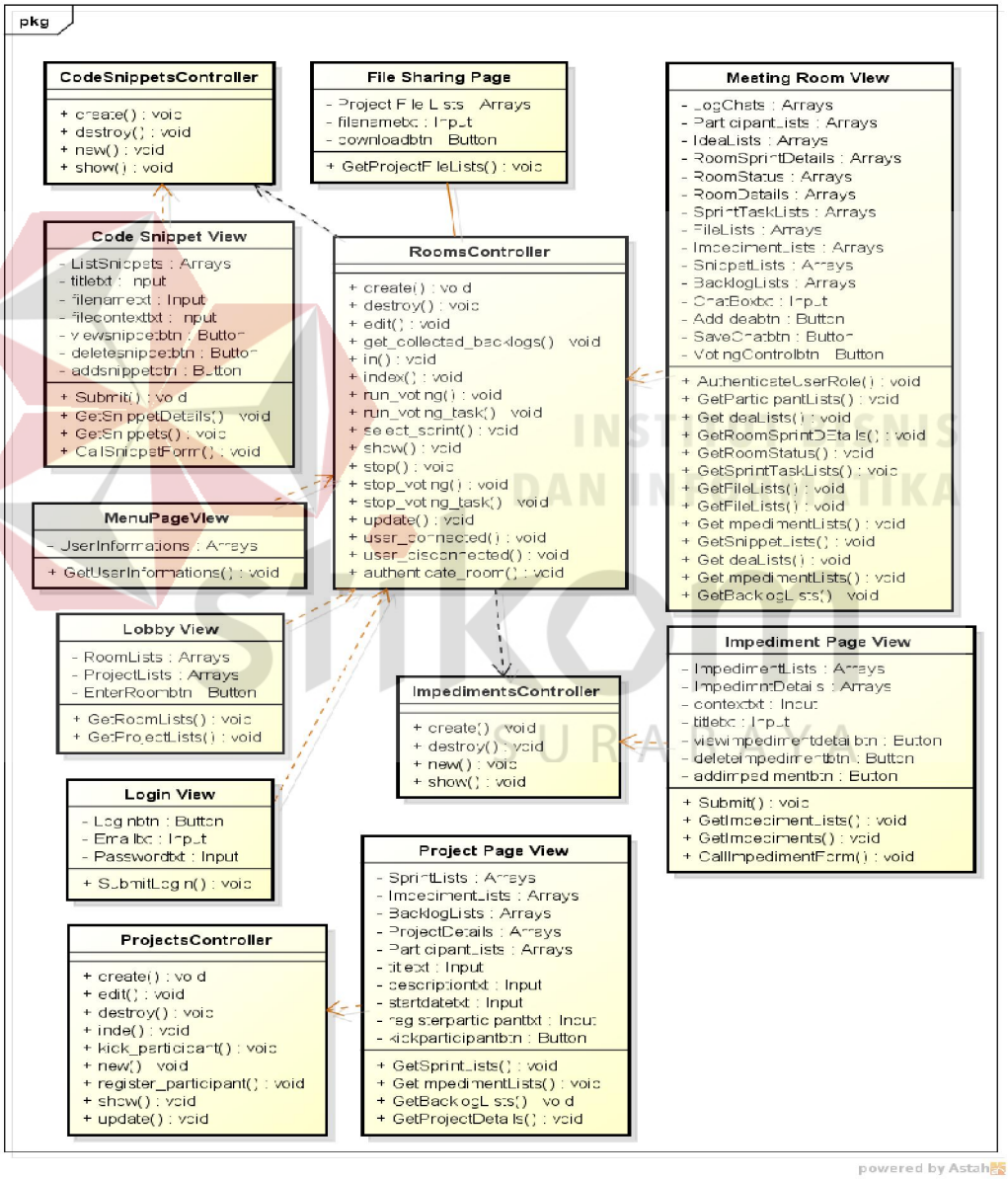
Gambar 3.57 *Class diagram* relasi antar *model*

Pada saat *controller* menerima *request* dari *client* yang membutuhkan fungsi untuk menampilkan data dari database, maka *controller* menghubungi *model* untuk melakukan fungsi tersebut. Hubungan antara *controller* dan *model* tergambar pada diagram kelas pada gambar 3.58



Gambar 3.58 *Class diagram* relasi antar *model-controller*

Controller bertindak sebagai *receiver* bagi *request* yang datang dari *view,* yang selanjutnya akan dihubungkan atau diteruskan kepada *back-end*. *Class diagram* hubungan antara *View-Controller* akan menggambarkan *request* dari *view* yang memerlukan *controller* untuk menanganinya. Diagram kelas hubungan antara *view-controller* dapat dilihat pada gambar 3.59 di bawah ini

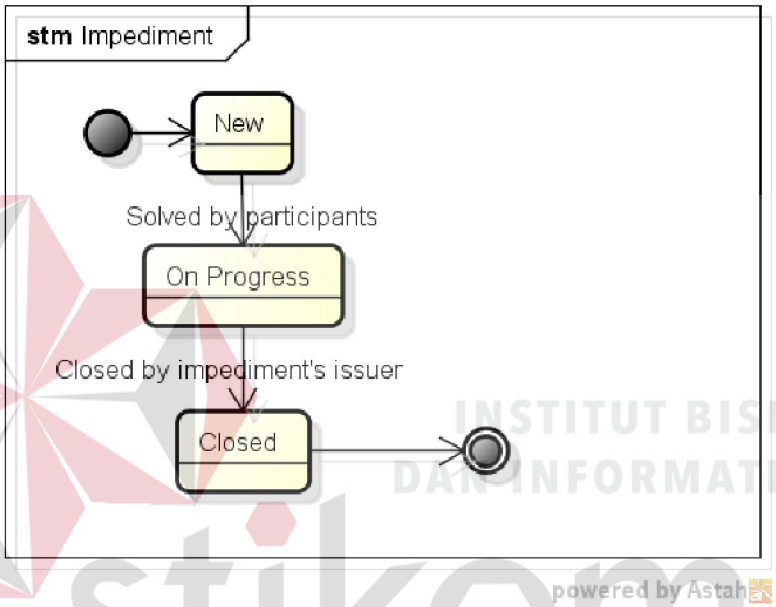


Gambar 3.59 *Class diagram* relasi *Controller-View*

3.3.9 *State Chart Diagram*

Selanjutnya, digambarkan *state chart diagram*, yang akan menjelaskan perubahan keadaan yang dialami oleh beberapa *object* yang ada pada aplikasi rapat online ini.

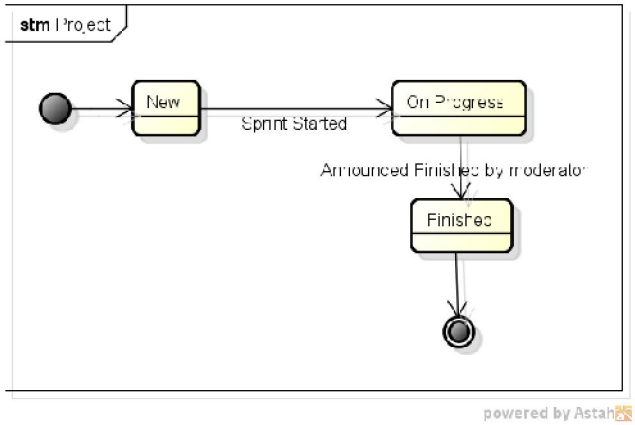
A. *State Chart Diagram impediment*



Gambar 3.60 *State Chart Diagram Impediment*

Pada Gambar 3.60, dijelaskan mengenai perubahan status dari *impediment*. Setiap peserta rapat, seperti telah disebutkan sebelumnya dapat membuat *impediment* pada ruangan rapat untuk didiskusikan dan diselesaikan bersama dengan kolaborasi. Setelah *impediment* dipecahkan, maka peserta dapat mengubah status *impediment* tersebut menjadi "*closed*".

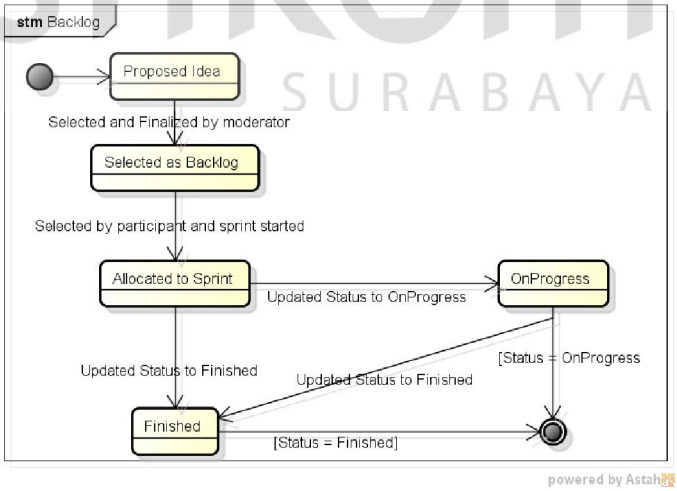
B. *State Chart Diagram sprint*



Gambar 3.61 *State Chart Diagram Project*

Pada Gambar 3.61, dijelaskan mengenai perubahan status dari *sprint*. Pada saat pertama *sprint* dibuat, maka status *sprint* akan secara otomatis menjadi *new*. Setelah melakukan rapat dan *sprint* dimulai, maka status *sprint* akan berubah menjadi *on progress*. Sedangkan pada akhir periode pengembangan, moderator akan menutup / mengakhiri *sprint*, yang akan merubah status *sprint* menjadi *done*.

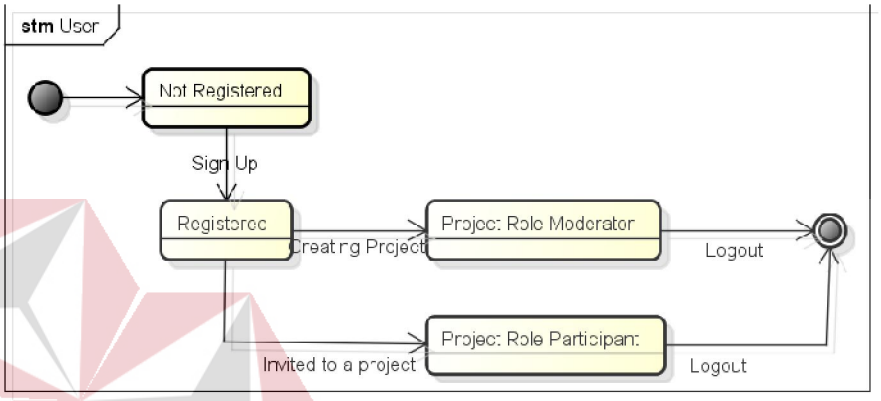
C. *State Chart Diagram backlog*



Gambar 3.62 *State Chart Diagram Backlog*

Pada Gambar 3.62, dijelaskan mengenai perubahan status dari *backlog* yang telah dialokasikan pada *sprint*. Pengubahan status *backlog* ini dapat dilakukan pada saat melakukan *daily meeting*, peserta dapat mengubah status *backlog* menjadi *on progress* atau *done*.

D. *State Chart Diagram user*

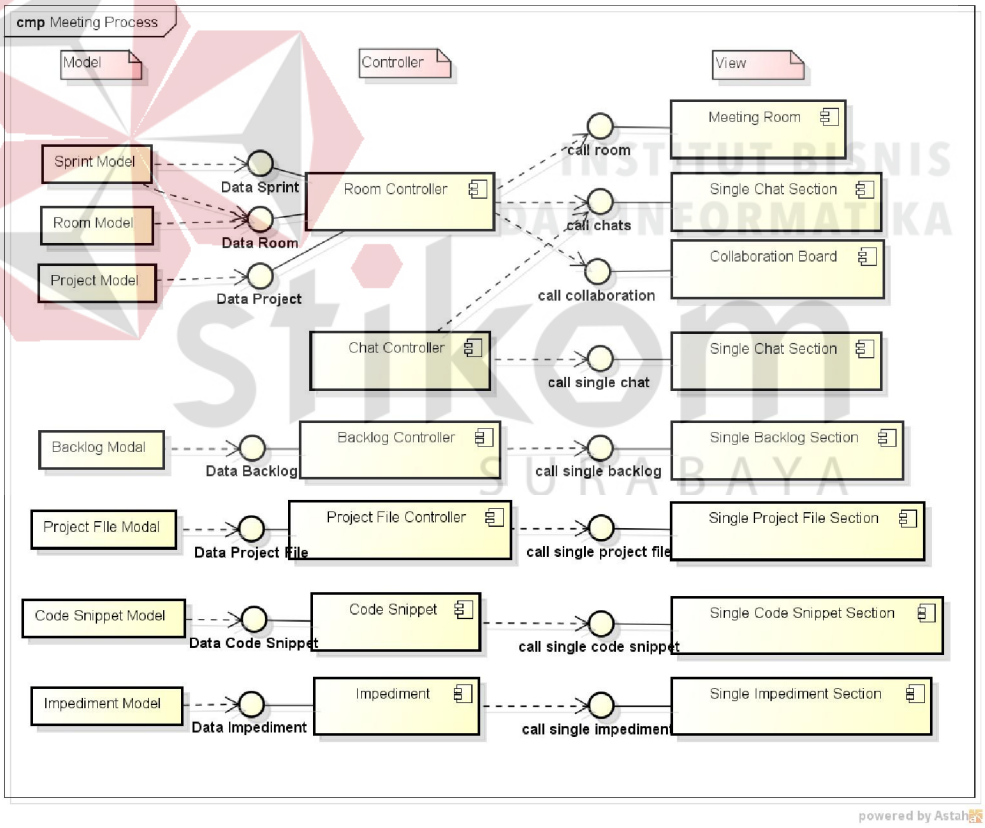


Gambar 3.63 *State Chart Diagram User*

Pada Gambar 3.63, dijelaskan mengenai perubahan status dari *backlog* yang telah dialokasikan pada *sprint*. Pengubahan status *backlog* ini dapat dilakukan pada saat melakukan *daily meeting*, peserta dapat mengubah status *backlog* menjadi *on progress* atau *done*.

3.3.10 *Component Diagram*

Pada diagram komponen, digambarkan bagaimana komponen dalam aplikasi saling berinteraksi. Komponen dari aplikasi rapat online ini adalah *model*, *view*, dan *controller* yang berinteraksi untuk melakukan beberapa aktivitas di dalam aplikasi. Penyediaan data yang dilakukan oleh *model* terhadap *controller,* digambarkan pada diagram komponen, begitupun dengan pemanggilan *view* oleh *controller* pada saat terjadinya suatu proses. Aktivitas yang akan digambarkan pada Gambar 3.64 ini adalah aktivitas utama di dalam aplikasi rapat online, yaitu aktivitas rapat.



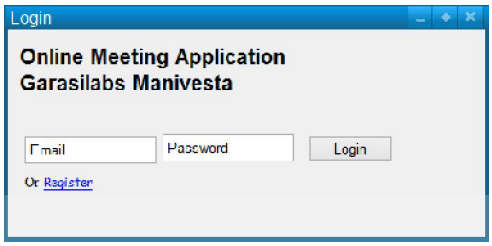
Gambar 3.64 *Component Diagram*

3.3.11 *Desain Interface*

Pada sub bab ini akan dibahas tentang desain *interface* yang akan dibuat untuk aplikasi rapat online pada PT. Garasilabs Manivesta.

A. Desain Interface Login

Desain *interface* ini digunakan untuk masuk ke dalam aplikasi.

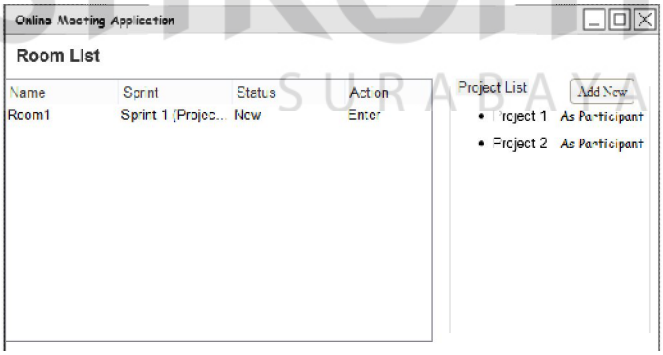


Gambar 3.65 Desain Interface Login

Di dalam halaman ini, terdapat dua input yaitu *username* dan *password* yang berguna untuk masukan autentikasi pada *database* bahwa *user* telah ada.

B. Desain Interface Lobby Page

Desain *interface* ini adalah *landing page* dari aplikasi

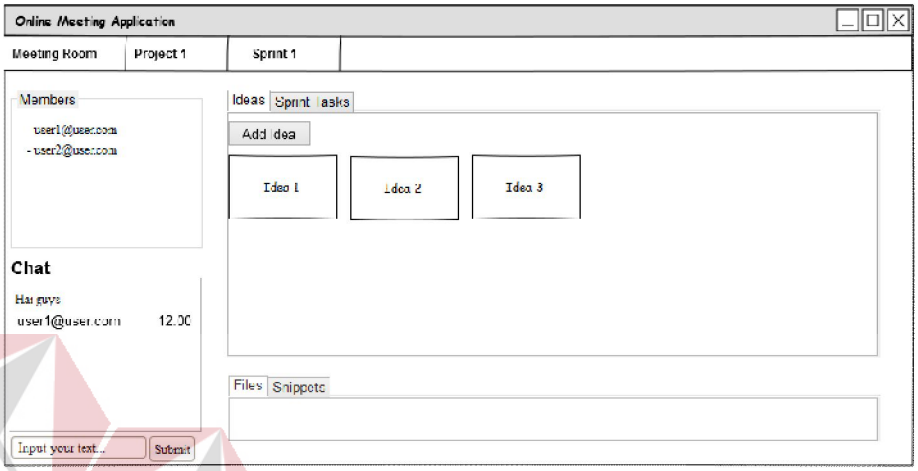


Gambar 3.66 Desain Interface Lobby Page

Pada halaman ini, ditampilkan daftar ruangan yang ada dan di sebelah kanan akan terdapat daftar project dimana *End-User* telah terdaftar.

C. Desain Interface Meeting Room

Desain *interface* ini adalah ruangan rapat yang akan digunakan di dalam aplikasi ini.

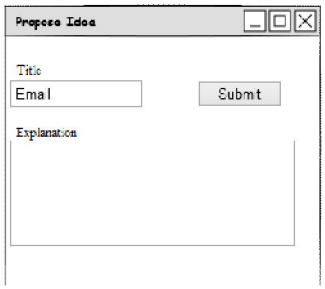


Gambar 3.67 Desain Interface Meeting Page

Pada halaman ini akan ditampilkan detil dari ruangan rapat yang ada pada bagian atas kiri halaman, yang terdiri dari nama *sprint, project*, dan ruangan. Selain itu, ditampilkan pula daftar peserta rapat yang telah masuk ke dalam ruangan. Di bagian kiri bawah terdapat juga bagian interaksi *chating* dari ruangan rapat. Pada bagian tengah atas, akan ditampilkan *section* untuk mengajukan ide-ide dalam rapat yang berkenaan dengan *project* serta melakukan voting, sedangkan bagian bawah terdapat *File Sharing* yang dapat digunakan peserta rapat untuk saling berbagi *file*.

D. Desain Interface Propose Idea

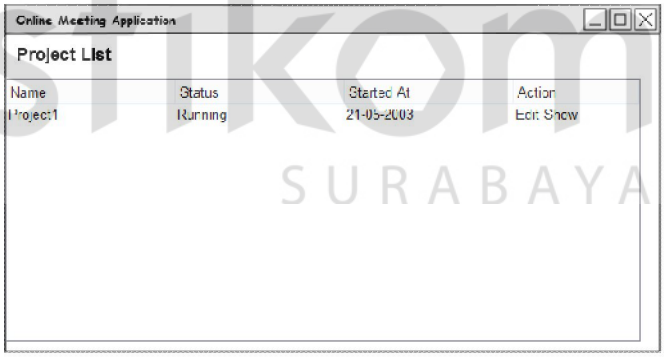
Desain *interface* ini adalah bagian dari halaman ruangan rapat yang merupakan *interface* untuk menambahkan / mengajukan ide. Nantinya *form* ini akan berupa *pop-up*.



Gambar 3.68 Desain Interface Propose Idea

E. Desain Interface Project Index Page

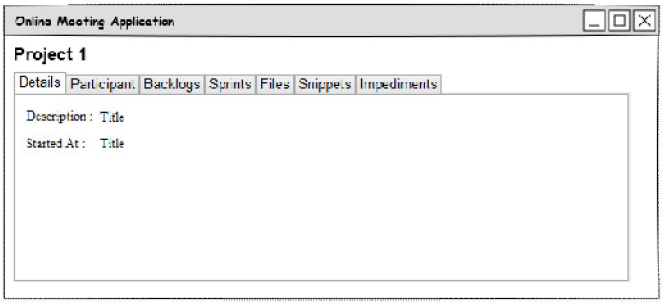
Desain *interface* ini menampilkan daftar *project* yang ada dimana *End-User* terdaftar di dalamnya.



Gambar 3.69 Desain Interface Project Index Page

F. Desain Interface Project Details Page

Desain *interface* ini akan menampilkan detil dari suatu *project*.

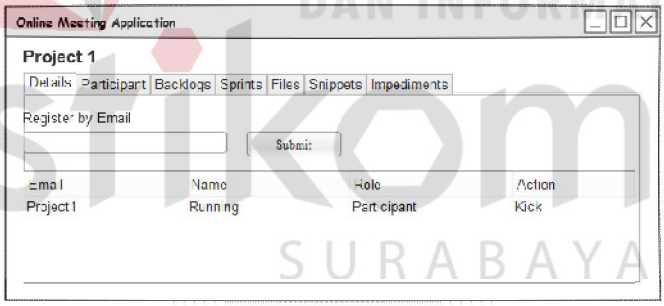


Gambar 3.70 Desain Interface Project Details Page

Terdapat juga navigasi berupa *tabbing* pada halaman ini yang dapat digunakan untuk melihat bagian-bagian dari suatu *project*, contohnya peserta rapat, *backlogs*, *sprints*, *files*, *snippets*, *impediments*.

G. Desain Interface Project Manage Participant

Desain *interface* ini akan menampilkan daftar peserta rapat dan juga *form* untuk menambahkan peserta rapat ke dalam *project*.

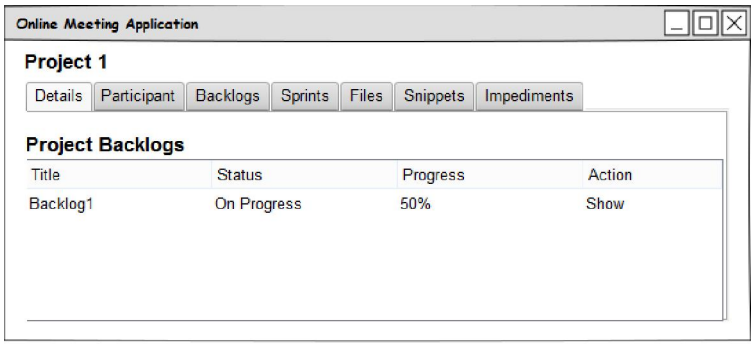


Gambar 3.71 Desain Interface Project Manage Participant

Pada daftar peserta juga diberikan informasi tentang *user role* dari peserta pada *project*. Jika yang melihat halaman ini adalah moderator, maka aksi *kick participant* dapat ditampilkan. Di sisi atas *tab content* akan terdapat *form* menambahkan peserta menggunakan *email* dari calon peserta.

H. Desain Interaface untuk Project Backlog List

Desain *interface* ini akan menampilkan daftar *backlogs* dari *project.*

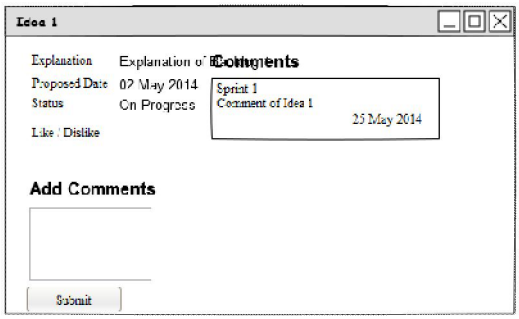


Gambar 3.72 Desain Interface Project Backlog List

Aplikasi akan menampilkan informasi di dalam bentuk tabel tentang status dari setiap *backlogs* dan perkembangannya (*progress*).

I. Desain Interface Idea Detail

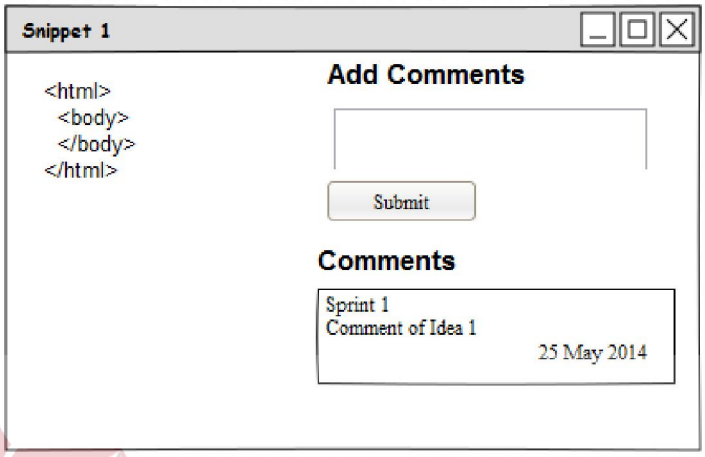
Interface ini merupakan *pop-up* yang ada pada setiap ide pada ruangan rapat. Aplikasi akan menampilkan detil dari ide yang dipilih beserta komentar-komentar yang telah diajukan sebelumnya. Pengguna dapat melakukan voting pada ide menggunakan halaman ini. Komentar yang berkenaan dengan ide juga dapat ditambahkan.



Gambar 3.73 Desain Interface Idea Detail

J. Desain Interface Code Snippets

Interface ini merupakan halaman yang menampilkan detil dari *code snippet*.

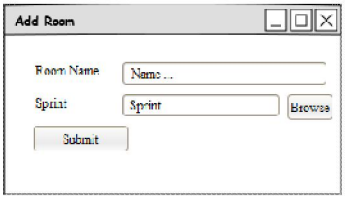


Gambar 3.74 Desain Interface Code Snippets

Pada halaman ini juga dapat dilihat pula komentar yang dimasukkan oleh pengguna yang berkenaan dengan *code snippet* tersebut.

K. Desain Interface Add Room Form

Interface ini merupakan halaman untuk menambahkan ruangan pada *lobby*. Disini pengguna akan memilih *sprint* yang akan dirapatkan di dalam ruangan rapat yang nantinya akan terbuat.

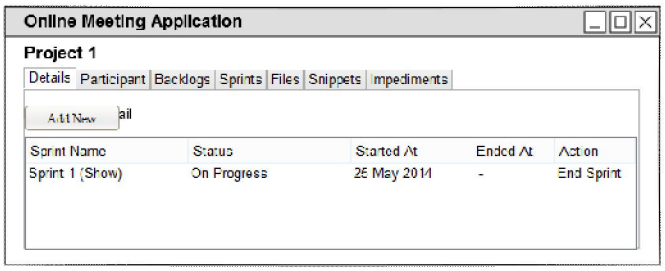


Gambar 3.75 Desain Interface Add Room Form

L. Desain Interface Manage Sprints

Interface ini merupakan bagian dari halaman detil *project* yang memberikan informasi tentang *sprint* yang telah dijalankan untuk

penyelesaian *project*. Disini juga ditampilkan informasi tentang status dari masing-masing *sprint*.

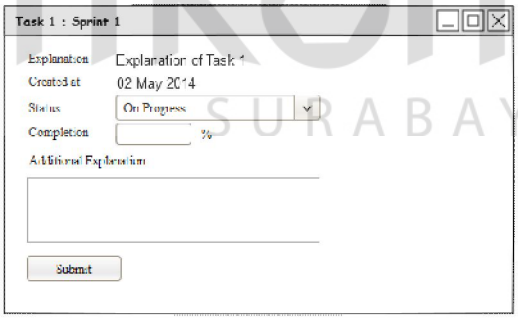


Gambar 3.76 Desain Interface Manage Sprints

Moderator juga dapat menambahkan *sprint*, setelah melakukan penyelesaian *sprint* yang berjalan.

M. Desain Interface Progres Reporting

Interface ini digunakan oleh pengguna untuk memasukkan perkembangan yang mereka telah lakukan terhadap masing-masing *task* dari *sprint*. Masukan yang diperlukan adalah *completion* dalam bentuk prosentase dan penjelasan tambahan (*additional explanation*).

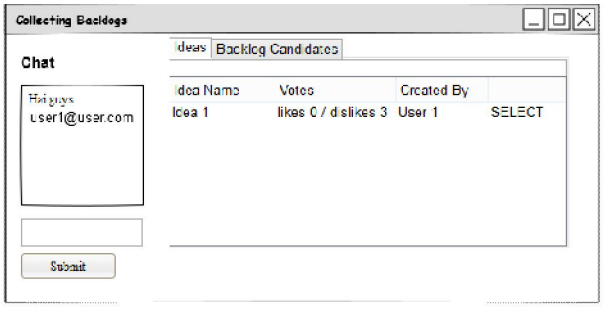


Gambar 3.77 Desain Interface Progress Reporting

N. Desain Interface Backlog Collecting

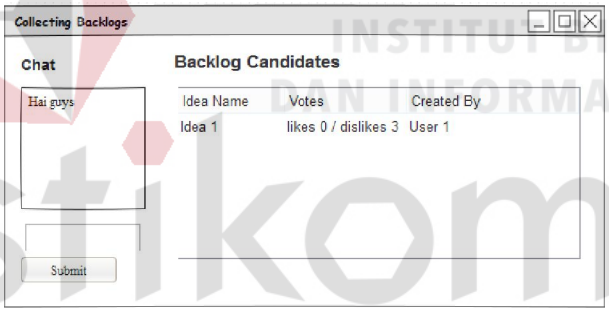
Interface untuk seleksi *backlogs* ini dibagi menjadi 2 bagian, sisi moderator dan sisi peserta rapat. Di sisi kiri halaman, terdapat *section chat,*

sehingga memungkinkan moderator dan peserta rapat melakukan interaksi seperti interupsi dan saran.



Gambar 3.78 Desain Interface Backlog Collecting sisi moderator

Pada desain *interface backlog collecting* sisi moderator, ditampilkan tombol "Select" yang dapat digunakan untuk memilih ide yang akan dimasukkan ke dalam kandidat *backlog*.

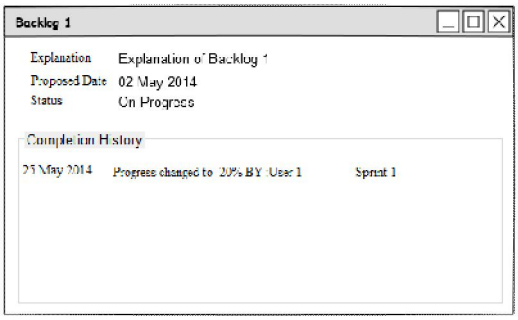


Gambar 3.79 Desain Interface Backlog Collecting sisi peserta

Pada desain *interface backlog collecting* sisi peserta, ditampilkan informasi ide yang telah dipilih oleh moderator menjadi kandidat *backlog*. Dengan ini, peserta dapat mengikuti aktivitas yang dilakukan oleh moderator.

O. Desain Interface Backlog Report Page

Dengan melakukan klik pada salah satu *backlogs* pada daftar *backlogs*, maka akan muncul *pop-up* yang berisi *history* perkembangan dari pengerjaan *backlog* tersebut seperti gambar di bawah ini.



Gambar 3.80 Desain Interface Backlog Updates Page

History akan ditampilkan beserta tanggal dan *sprint* dimana aktivitas yang berkenaan dengan *backlog* tersebut terjadi. Disini dapat dilihat laporan penyelesaian dari suatu *backlog*.

3.3.12 Rancangan Uji Coba *User Acceptance*

Rancangan uji coba ini digunakan untuk merancang pernyataan – pernyataan yang nantinya akan digunakan sebagai pengukur kelayakan aplikasi.. Adapun pembobotan yang digunakan pada saat penilaian dapat dilihat pada Tabel 3.26.

Tabel 3.26 Tolak ukur penilaian pada *user acceptance test*

No	Tolak Ukur Penilaian	Bobot	Keterangan
1	*Strongly Agree*	5	Sangat Setuju dengan pernyataan yang diajukan
2	*Agree*	4	Hanya setuju dengan pernyataan yang diajukan
3	*Doubtful*	3	Meragukan pernyataan yang diajukan
4	*Disagree*	2	Tidak setuju akan pernyataan yang diajukan
5	*Strongly Disagree*	1	Menolak pernyataan yang diajukan

Pernyataan – pernyataan di dalam angket lalu akan dibuat berdasarkan 3 aspek penting yang ingin diukur pada aplikasi rapat online ini Beberapa aspek penting itu dapat dilihat pada Tabel 3.27.

Tabel 3.27 Aspek utama pengukuran dalam *user acceptance*

No	Aspek Utama pengukuran aplikasi
1	Kemudahan penggunaan pertama kali.
2	Kemudahan menggunakan fitur utama aplikasi
3	Kesesuaian aplikasi dengan kebutuhan pengguna.

A. Kemudahan penggunaan pertama kali

User acceptance secara langsung membahas aspek hal-hal yang berkaitan dengan panduan pengguna pada saat pertama menggunakan aplikasi dan kemampuan aplikasi dari segi *interface* untuk dapat diadaptasi oleh pengguna dalam kurun waktu tertentu. Pernyataan untuk aspek ini, dapat dilihat pada Tabel 3.28

Tabel 3.28 Rancangan pengukur kemudahan penggunaan pertama kali

No	Pernyataan
1	*For the first time use, this application is quite easy to be learned*
2	*There are first time use helpers in this application*
3	*The first time use helper in this application is helpful*
4	*You took a few times to getting used to in using this application*

B. Kemudahan menggunakan fitur utama aplikasi

User acceptance membahas mengenai tingkat kemudahan aplikasi untuk dipahami oleh pengguna. Beberapa hal yang diukur adalah penggunakan aplikasi, baik dari alur penggunaan dan peletakan elemen - elemen fungsi dari setiap modul. Pernyataan untuk aspek ini, dapat dilihat pada Tabel 3.29

Tabel 3.29 Rancangan pengukur kemudahan menggunakan fitur utama aplikasi

No	Pernyataan
1	*You, as the meeting participant, not confused with these moderation tools*
2	*The collecting backlogs layout well displayed*
3	*Proposing your ideas is easy for you*
4	*During the meeting, it is not hard to find the moderation button, like button to run/stop voting for task and proposing idea*
5	*Reporting your progresses is easy for you*

C. Kesesuaian aplikasi dengan kebutuhan pengguna

User acceptance ini bertujuan menguji apakah aplikasi rapat online ini dapat memfasilitasi kegiatan rapat *scrum* pada perusahaan dan memenuhi kebutuhan pengguna aplikasi. . Pernyataan untuk aspek ini, dapat dilihat pada Tabel 3.30

Tabel 3.30 Rancangan pengukur kesesuaian aplikasi dengan kebutuhan pengguna

No	Pernyataan
1	*There are many betterments of the online meeting processes using this application, compared with Yahoo Messanger*
2	*The application facilitates the scrum meeting*
3	*This application help you to do the coding documentation*
4	*If you have any impediments on a project, the application impediment feature can help you to collaborate with the others to solve it*
5	*You do not often find an error or dysfunctional feature when you were on the meeting room*
6	*The monitoring feature for each sprint is very helpful to track sprint progresses*