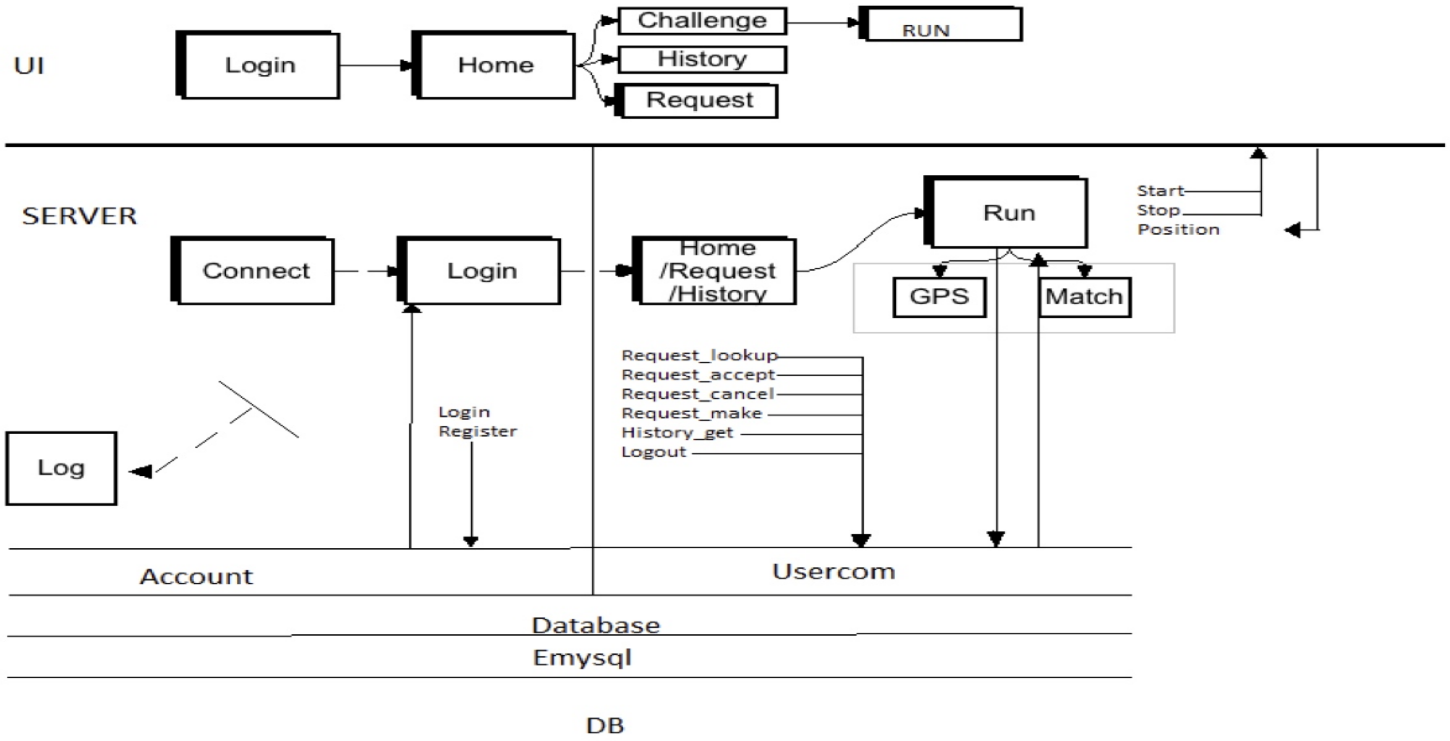
Datarace

Don’t count the race, make the races count.

Introduktion

- En löpar-app.
- Demonstration

Systemarkitektur



Organisation & Planering

- Trello - planering av veckomål
- GitHub - koddistribution
- Dagligt gruppmöte

Concurrency

- Både app och server använder concurrency
- Server - message passing
- Appen - använder ramverk som övervakar concurrencyn, inga behov av lås.

Exempel

```
NSOperationQueue *myQueue = [[NSOperationQueue alloc] init];
    [myQueue addOperationWithBlock:^{
```

This is where the magic happens

```
[[NSOperationQueue mainQueue] addOperationWithBlock:^{
```

Tillbaka till main queue

```
    }
};
```

Concurrency-implementation

Fördelar

Enkelt att använda

Lätt att övervaka

Nackdelar

Svårt att speca(appen)

DEMO

DEMO!!

Guldkorn

Nätverkskommunikation och meddelandehantering på servern.

```
%% Från listener_sup.erl

init({Port, Listeners}) ->
    {ok, ListenSocket} = gen_tcp:listen(Port, [binary,
                                               inet,
                                               {reuseaddr, true},
                                               {active, once},
                                               {packet, 4}]),
    SuperSpec = {simple_one_for_one, 100, 500},
    ChildSpec = {listener,
                 {listener, start_link, [ListenSocket]},
                 transient, 5000, worker, [listener]},
    spawn_link(?MODULE, start_listeners, [Listeners]),
    {ok, {SuperSpec, [ChildSpec]}}.
```

```
%% Från client_serv.erl

init(_Args) ->
    process_flag(trap_exit, true),
    LoopData = #loop_data{},
    {ok, verify, LoopData}.

...

main(?LOGIN_LOGOUT, LoopData) ->
    {stop, normal, LoopData};
main({?REQUEST,
      <<ChallengeId:32/little-integer,
        Distance:32/little-integer>>},
     LoopData) ->
...

match({?MATCH_GPS, Packet}, LoopData) ->
...
```

```
%% Fortsättning client_serv ...

handle_info({tcp, _, <<Type:2/binary>>}, State, LoopData) ->
    inet:setopts(LoopData#loop_data.socket, [{active, once}]),
    ?MODULE:State(Type, LoopData);

handle_info({tcp, _, <<Type:2/binary, Packet/binary>>}, State, LoopData) ->
    inet:setopts(LoopData#loop_data.socket, [{active, once}]),
    ?MODULE:State({Type, Packet}, LoopData);

handle_info({tcp_closed, _}, _State, LoopData) ->
    log_serv:log("User disconnected unexpectedly"),
    {stop, normal, LoopData};

handle_info({tcp_error, _, _Reason}, _State, LoopData) ->
    log_serv:log("Tcp error"),
    {stop, normal, LoopData}.

terminate(_Reason, State, LoopData) ->
    account:logout(LoopData#loop_data.user_id),
    log_serv:log("Logged out user"),
    log_serv:log("Terminating client server").
```

Ruttna ägg…

- Appen hade småbuggar som kunnat undvikas om vi varit mer vana med språket.
- Utloggning vid server stop.
- Skulle kunna utökat till n spelare per race.
- Delat upp db kod och logik mer.