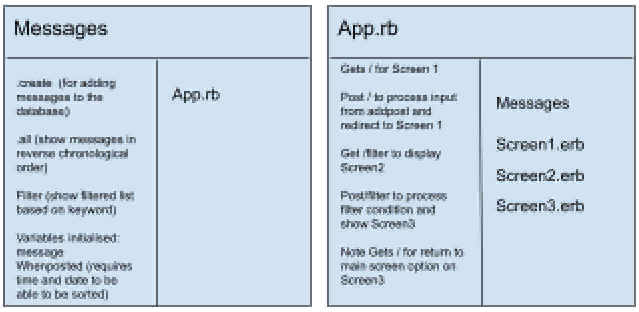
CRC cards

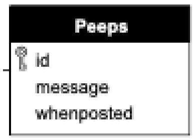


Messages	
.create (for adding messages to the database) .all (show messages in reverse chronological order) Filter (show filtered list based on keyword) Variables initialised: message Whenposted (requires time and date to be able to be sorted)	App.rb

App.rb	
Gets / for Screen 1 Post / to process input from addpost and redirect to Screen 1 Get /filter to display Screen2 Post/filter to process filter condition and show Screen3 Note Gets / for return to main screen option on Screen3	Messages Screen1.erb Screen2.erb Screen3.erb

Entity diagram:

Note whenposted to be TIMESTAMP WITH TIME ZONE DEFAULT CURRENT_TIMESTAMP



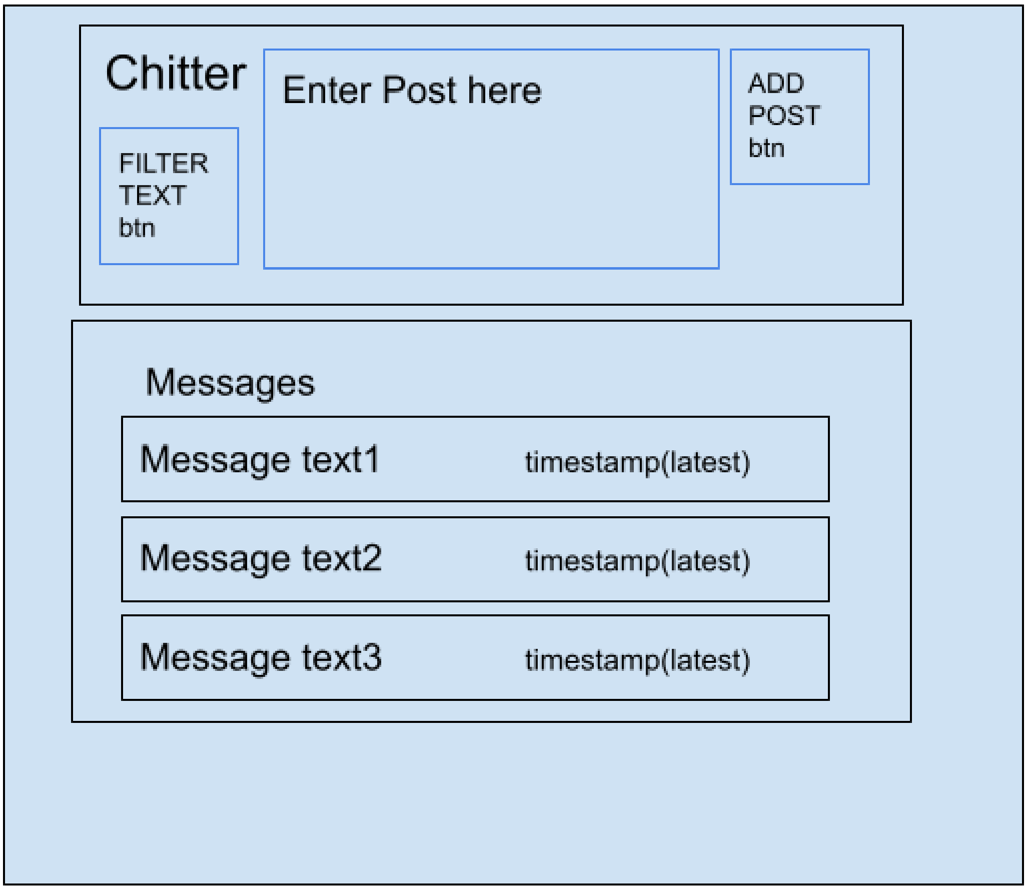
Peeps
id
message
whenposted

```
ALTER TABLE peeps
        ADD whenposted TIMESTAMP WITH TIME ZONE DEFAULT
CURRENT_TIMESTAMP;
```

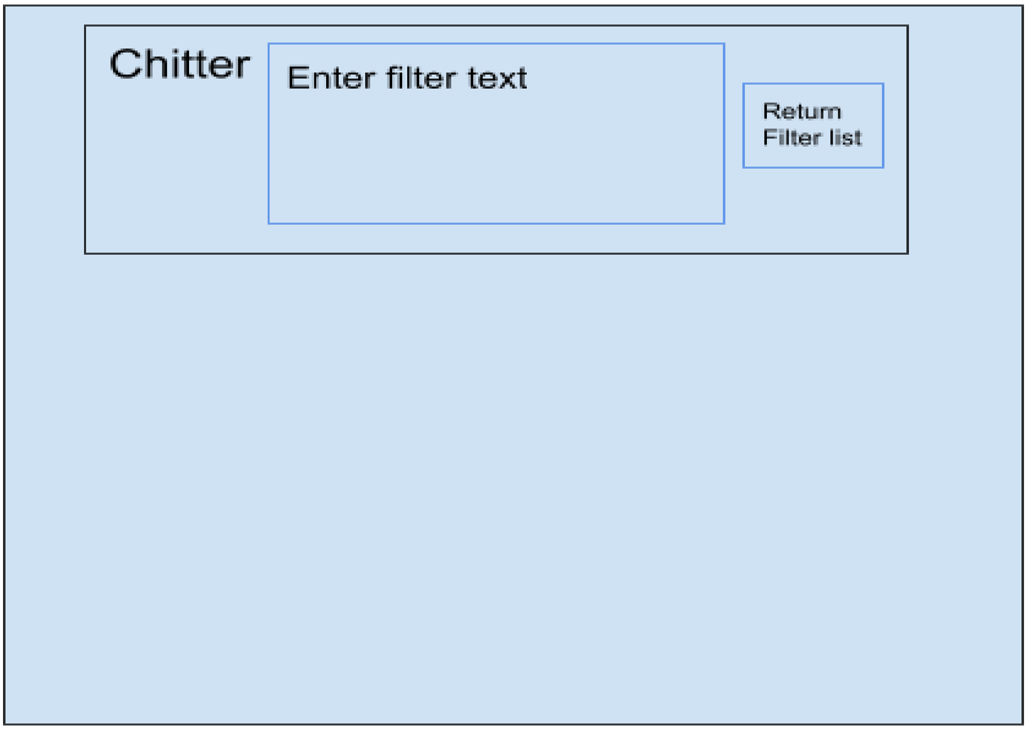
Note assuming that the user will supply their name in the message field and this will be filterable, as user stories have not specified a name explicitly.

Sketch of each page of app that user sees

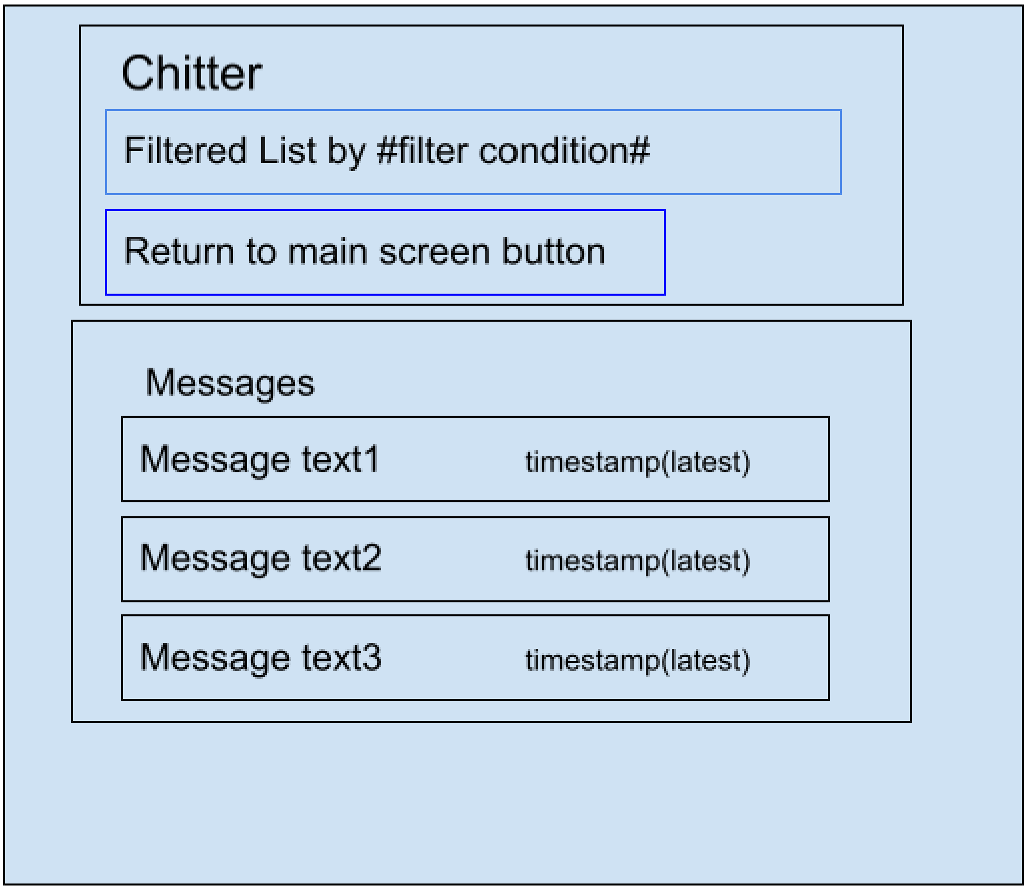
Screen1



Screen2



Screen3


Chitter

Filtered List by #filter condition#

Return to main screen button

Messages

Message text1 timestamp(latest)

Message text2 timestamp(latest)

Message text3 timestamp(latest)

User stories.

As a Maker
So that I can see what people are doing
I want to see all the messages (peeps)
in a browser
```
As a Maker
So that I can let people know what I am doing
I want to post a message (peep) to chitter
```

```
As a Maker
So that I can see when people are doing things
I want to see the date the message was posted
```
(Hint the database table will need to change to store the date too)

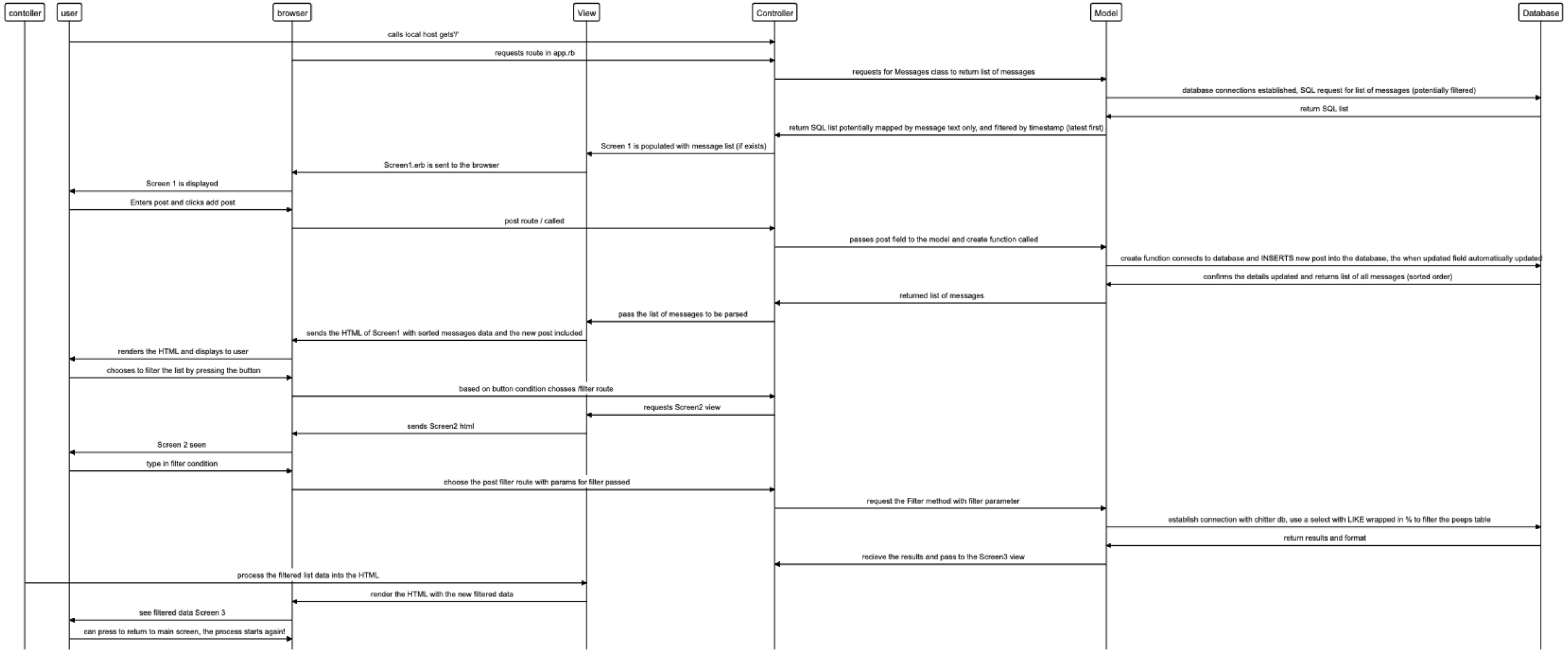
```
As a Maker
So that I can easily see the latest peeps
I want to see a list of peeps in reverse chronological order
```
```
As a Maker
So that I can find relevant peeps
I want to filter on a specific keyword

Nouns: messages(peeps), chitter, date, keyword
Verbs: see, post, list, order, filter
```

List of requests / responses involved (word + diagram)

```
alias user="user"
alias  browser = "browser"
alias view="View"
alias controller = "Controller"
alias model = "Model"
alias database = "Database"


user -> controller: "calls local host gets'/'"
browser -> controller: "requests route in app.rb"
controller -> model: "requests for Messages class to return list of messages"
model -> database: "database connections established, SQL request for list of messages (potentially filtered)"
database -> model: "return SQL list"
model -> controller: "return SQL list potentially mapped by message text only, and filtered by timestamp (latest first)"
controller -> view: "Screen 1 is populated with message list (if exists)"
view -> browser: "Screen1.erb is sent to the browser"
browser -> user: "Screen 1 is displayed"
user -> browser: "Enters post and clicks add post"
browser -> controller: "post route / called"
controller -> model: "passes post field to the model and create function called"
model -> database: "create function connects to database and INSERTS new post into the database, the when updated field automatically updated"
database -> model: "confirms the details updated and returns list of all messages (sorted order)"
model -> controller: "returned list of messages"
controller -> view: "pass the list of messages to be parsed"
view -> browser: "sends the HTML of Screen1 with sorted messages data and the new post included"
browser -> user: "renders the HTML and displays to user"
user -> browser: "chooses to filter the list by pressing the button"
browser -> controller: "based on button condition chooses /filter route"
controller -> view: "requests Screen2 view"
view -> browser: "sends Screen2 html"
browser -> user: "Screen 2 seen"
user -> browser: "type in filter condition"
browser -> controller: "choose the post filter route with params for filter passed"
controller -> model: "request the Filter method with filter parameter"
model -> database: "establish connection with chitter db, use a select with LIKE wrapped in % to filter the peeps table"
database -> model: "return results and format"
model -> controller: "receive the results and pass to the Screen3 view"
controller -> view: "process the filtered list data into the HTML"
view -> browser: "render the HTML with the new filtered data"
browser -> user: "see filtered data Screen 3"
user -> browser: "can press to return to main screen, the process starts again!"
```


contoller
user
browser
View
Controller
Model
Database
calls local host gets'/'
requests route in app.rb
requests for Messages class to return list of messages
database connections established, SQL request for list of messages (potentially filtered)
return SQL list
return SQL list potentially mapped by message text only, and filtered by timestamp (latest first)
Screen 1 is populated with message list (if exists)
Screen1.erb is sent to the browser
Screen 1 is displayed
Enters post and clicks add post
post route / called
passes post field to the model and create function called
create function connects to database and INSERTS new post into the database, the when updated field automatically updated
confirms the details updated and returns list of all messages (sorted order)
returned list of messages
pass the list of messages to be parsed
sends the HTML of Screen1 with sorted messages data and the new post included
renders the HTML and displays to user
chooses to filter the list by pressing the button
based on button condition chosses /filter route
requests Screen2 view
sends Screen2 html
Screen 2 seen
type in filter condition
choose the post filter route with params for filter passed
request the Filter method with filter parameter
establish connection with chitter db, use a select with LIKE wrapped in % to filter the peeps table
return results and format
recieve the results and pass to the Screen3 view
process the filtered list data into the HTML
render the HTML with the new filtered data
see filtered data Screen 3
can press to return to main screen, the process starts again!

NOTE a whenposted field is assumed in the diagram above: