# Meeting 22/8

Present: Jon, Aran, Martin, Stephen

Items Discussed:

* Generate Epic Stories
  + Decided on 4-5 epic stories
  + Breaking down into user stories to be divided
* Logbook
  + To be done on google docs, minutes uploaded to git
* Everyone allocated 1-2 epic stories
* Classes
  + Some can be done before Use Cases/User story are finalised
  + Jon and Martin to develop basic classes
  + Everyones got allocated use cases to work ons

Next Meetings: Wednesday 29/08 – 3pm

* Finalise User Stories
* Elaborate on UML diagram
* Begin implantation planning

# Meeting 29/8

Present: Jon, Martin, Stephen, Nat, Aran

Items:

1. User Stories
   * Need refining - everyone to critique others to get feedback
   * Need to be atomic, if there’s an option to split it, then split it
   * Specific acceptance criteria, need to be able to complete a module, and once it ticks of the AC then it is done, some are really vague
   * Distinguish between AC and new features
   * Features that aren’t necessarily in the brief explicitly…

Added new user stories regarding login details, viewing profiles

Everyone to go through and add comments to other user stories

Add acceptance criteria for all

Everyone to complete by Thursday/Friday

1. Classes
   * Start doing class diagrams
   * At least an early concept, have it done early to get feedback
2. Implementation
   * When do we want to start implementation (next week?)
   * Division of work…
3. Other notes
   * Repo organisation - classes in folders etc…?
   * Redefine User Points system? Make a bit smaller
   * Style convention - be consistent across everyones code

module\_name, package\_name, ClassName, method\_name, ExceptionName, function\_name, GLOBAL\_CONSTANT\_NAME, global\_var\_name, instance\_var\_name, function\_parameter\_name, local\_var\_name (for example…. From stackexchange/PEP8)

1. Deadlines
   * User stories - Sunday
   * Class diagram - Wk 9?? - to be done well before
   * First release - Wk 9

# Feedback 29/8

Feedback on user stories from laboratory session

* Need to be more specific with benefits. Some of them just restate a function, need to be a genuine benefit to the customer (almost like a selling point)
* Need to be a lot more specific with assessment criteria. Go into as much detail as possible
* Condense epic stories. Go from 8-> 3ish
* We had some features that might be nice, but aren’t in the spec, so lets not create extra work for ourselves
* Group to go back and adjust user stories, condense into new epic stories
* Deadline by Friday to be reviewed by Saturday and submitted on Sunday

# Meeting 5/9

Present: Jon, Aran, Martin, Stephen, Nat

1. CRC Cards
   * Basic drafting of CRC cards
   * Need to make sure class methods are for its own attributes
   * Discussed where to put searches -> within “manager” classes
   * Appointment Manager - universal list of appointments, indexed by ID
2. Class Diagram
   * Translation of CRC cards - Basic draft done by Jon/Martin
   * Need to finalise with CRC cards
   * Need to clear a few things up with tutors - getter/setter methods in class diagram etc..?
3. Implementation Plan
   * Need to find a way to get consistency across functions - will do this by specifying method outputs on UML
   * Options - split classes up per person, have to wait for functions to finish

* Allow concurrent access to classes, add functions we need for

1. User Stories
   * Login:

*3.1 - Login Successfully (2 pts)*

Classes - User, User Manager, Patient, Provider

Responsible: Aran

* + Search:

*1.1 - Search for Centre (4 pts)*

Classes: Centre, CentreManager, SystemManager

Responsible: Jon

* + 1. *1.2 - Search for Service (2 pts)*

Classes: Provider, UserManager, SystemManager

Responsible:Jon

* + 1. *1.3 - Search for providers (2 pts)*

Classes: Provider, UserManager, SystemManager

Responsible:

* + View profiles:

*1.4 - View provider profile (4 pts)*

Classes: Provider, SystemManager

Responsible: Martin

* + 1. *1.5 - View center profile (4 pts)*

Classes: Centre, SystemManager

Responsible: Martin

* + Book:

*1.6 - Book an appointment (4 pts)*

Classes: Patient, Provider, Centre, AppointmentManager

Responsible: Nat

* + View Appts:

*2.2 - Patient view current appointments (4 pts)*

Classes: Patient, AppointmentManager

Responsible: Stephen

* + 1. *2.6 - Provider view current appointments (4 pts)*

Classes: Provider, AppointmentManager

Responsible: Stephen

5. Classes

User, user manager = Aran

Patient, provider = Nat

Centre, centre manager = Jon

Appointment, Appointment manager

# Meeting 10/09

Present: Aran, Nat, Jon

1. Finalise classes allocation
   * Completed skeleton for most classes, just need to write appointment class
   * Jon to do??
2. Complete routes/templates
   * This will probably be the hard part to get used to
   * Python functions shouldn’t take long, need to work on how to display to html
3. Unit tests
   * Everyone to do testing documents on their own branch
   * Using pytest framework
4. Directory management
   * How we want to organise files
   * Directories for model, routes, templates, static, tests etc…
   * Anything
   * Discuss in tute about how to implement