4 Gemfile View

```diff
@@ -44,12 +44,12 @@ group :production, :test, :replica do
 end
 
 group :development, :test, :replica do
-  gem 'rspec-rails', '~> 3'
   gem 'factory_girl_rails'
   gem 'minitest-rails'
   gem 'minitest-hyper'
   gem 'database_cleaner'
-  gem "minitest-osx"
+  gem 'minitest-osx'
+  gem 'minitest-around'
 end
 
 # Student submission
```

22 Gemfile.lock View

```diff
@@ -81,7 +81,6 @@ GEM
     devise_ldap_authenticatable (0.8.5)
       devise (>= 3.4.1)
       net-ldap (>= 0.6.0, <= 0.11)
-    diff-lcs (1.2.5)
     docile (1.1.5)
     encryptor (1.3.0)
     enumerable-lazy (0.0.1)
@@ -129,6 +128,8 @@ GEM
     mime-types-data (3.2016.0521)
     mini_portile2 (2.1.0)
     minitest (5.9.0)
+    minitest-around (0.3.2)
+      minitest (~> 5.0)
     minitest-hyper (0.2.0)
     minitest-osx (0.1.0)
       minitest (~> 5.4)
@@ -199,23 +200,6 @@ GEM
     responders (2.2.0)
       railties (>= 4.2.0, < 5.1)
     rmagick (2.15.4)
-    rspec-core (3.4.4)
-      rspec-support (~> 3.4.0)
-    rspec-expectations (3.4.0)
-      diff-lcs (>= 1.2.0, < 2.0)
-      rspec-support (~> 3.4.0)
-    rspec-mocks (3.4.1)
-      diff-lcs (>= 1.2.0, < 2.0)
-      rspec-support (~> 3.4.0)
-    rspec-rails (3.4.2)
-      actionpack (>= 3.0, < 4.3)
-      activesupport (>= 3.0, < 4.3)
-      railties (>= 3.0, < 4.3)
-      rspec-core (~> 3.4.0)
-      rspec-expectations (~> 3.4.0)
-      rspec-mocks (~> 3.4.0)
-      rspec-support (~> 3.4.0)
-    rspec-support (3.4.1)
     ruby-filemagic (0.7.1)
     ruby-progressbar (1.8.1)
```

```
      rubyzip (1.2.0)
@@ -273,6 +257,7 @@ DEPENDENCIES
   grape-swagger
   hirb
   launchy
+  minitest-around
   minitest-hyper
   minitest-osx
   minitest-rails
@@ -287,7 +272,6 @@ DEPENDENCIES
   rails_best_practices
   require_all (= 1.3.3)
   rmagick (~> 2.15)
-  rspec-rails (~> 3)
   ruby-filemagic
   rubyzip
   simplecov
```

6 app/api/api.rb View

```
@@ -32,6 +32,9 @@ class Root < Grape::API
     mount Api::Submission::PortfolioApi
     mount Api::Submission::PortfolioEvidenceApi
     mount Api::Submission::BatchTask
+    mount Api::Helpdesk::Ticket
+    mount Api::Helpdesk::Session
+    mount Api::Helpdesk::Stats

     #
     # Add auth details to all end points
@@ -51,6 +54,9 @@ class Root < Grape::API
     AuthHelpers.add_auth_to Api::Submission::PortfolioApi
     AuthHelpers.add_auth_to Api::Submission::PortfolioEvidenceApi
     AuthHelpers.add_auth_to Api::Submission::BatchTask
+    AuthHelpers.add_auth_to Api::Helpdesk::Ticket
+    AuthHelpers.add_auth_to Api::Helpdesk::Session
+    AuthHelpers.add_auth_to Api::Helpdesk::Stats

     add_swagger_documentation \
       base_path: nil,
```

104 app/api/helpdesk/session.rb View

```
@@ -0,0 +1,104 @@
+require 'grape'
+
+module Api
+  module Helpdesk
+    #
+    # Helpdesk session system endpoints
+    #
+    class Session < Grape::API
+      helpers AuthHelpers
+      helpers AuthorisationHelpers
+      helpers LogHelper
+
+      before do
+        authenticated?
+      end
+
+     # ------------------------------------------------------------------------
+     # GET /helpdesk/sessions?user_id=[id]&is_active=[true|false]
+     # ------------------------------------------------------------------------
+     desc "Get helpdesk sessions"
+     params do
+       optional :user_id,   type: Integer, :desc => "Filter by specific user id"
+       optional :is_active, type: Boolean, :desc => "Filter to only active sessions"
+     end
+     get '/helpdesk/sessions' do
+       unless authorise? current_user, HelpdeskSession, :get_sessions
+         error!({"error" => "Not authorised get helpdesk sessions"}, 403)
+       end
+       if params.empty?
+         logger.info "#{current_user.username} requested all helpdesk sessions"
+         HelpdeskSession.all
+       else
+         user_id = params[:user_id]
+         is_active = params[:is_active]
+         logger.info "#{current_user.username} requested all #{is_active ? 'active' : ''} helpdesk sessions" << (!user
+         sessions = is_active ? HelpdeskSession.active_sessions : HelpdeskSession.all
+         sessions = user_id.nil? ? sessions : sessions.where(user_id: user_id)
+         sessions
+       end
+     end
+
+     # ------------------------------------------------------------------------
+     # POST /helpdesk/sessions
+     # ------------------------------------------------------------------------
+     desc "Begin a new session at the helpdesk as current user"
+     params do
+       requires :clock_off_time, type: DateTime, :desc => "The estimated clock off time when the user ends working"
+       optional :user_id, type: Integer, :desc => "The user who is about to begin working (if blank, posting user is u
+     end
+     post '/helpdesk/sessions' do
+       unless authorise? current_user, HelpdeskSession, :create_session
+         error!({"error" => "Not authorised to create a helpdesk session"}, 403)
+       end
+       user = params[:user_id] ? User.find(params[:user_id]) : current_user
+       # TODO: if current_user != user_id passed in then require PIN
+       if HelpdeskSession.user_clocked_off?(user)
+         session = HelpdeskSession.create!(
+           user: user,
+           clock_on_time: DateTime.now, # Clock on immediately when ticket is created
+           clock_off_time: params[:clock_off_time]
+         )
+         logger.info "#{current_user.username} created new helpdesk session (id=#{session.id})"
+       else
+         error!({"error" => "#{user.username} is already clocked on at the helpdesk"}, 403)
+       end
+       session
+     end
+
+     # ------------------------------------------------------------------------
+     # DELETE /helpdesk/sessions/:id
+     # ------------------------------------------------------------------------
+     desc "Prematurely clock off an existing helpdesk session"
+     params do
+       requires :id, type: Integer, :desc => "The session to clock off"
+     end
+     delete '/helpdesk/sessions/:id' do
+       session = HelpdeskSession.find(params[:id])
+       unless authorise? current_user, session, :clock_off_session
+         error!({"error" => "Not authorised to clock off helpdesk session (id=#{session.id})"}, 403)
+        end
+        # TODO: if current_user != session.user then require PIN
+        if session.clocked_off?
+          logger.info "#{current_user.username} attempted to clock off already clocked off helpdesk session (id=#{sessi
+        else
+          session.clock_off
+          logger.info "#{current_user.username} prematurely clocked off helpdesk session (id=#{session.id})"
+        end
+        session
+      end
+
+      # ------------------------------------------------------------------------
+      # GET /helpdesk/sessions/tutors
+      # ------------------------------------------------------------------------
+      desc "Get a list of all currently tutors working at the helpdesk"
+      get '/helpdesk/sessions/tutors' do
+        unless authorise? current_user, HelpdeskSession, :get_all_current_session_users
+          error!({"error" => "Not authorised view current helpdesk staff"}, 403)
+        end
+        logger.info "#{current_user.username} requested all currently working helpdesk staff"
+        ActiveModel::ArraySerializer.new(HelpdeskSession.users_working_now, each_serializer: HelpdeskUserSerializer)
+      end
+    end
+  end
+end
```

76 app/api/helpdesk/stats.rb View

```
@@ -0,0 +1,76 @@
+require 'grape'
+
+module Api
+  module Helpdesk
+    #
+    # Helpdesk ticket stats endpoints
+    #
+    class Stats < Grape::API
+      helpers AuthHelpers
+      helpers AuthorisationHelpers
+      helpers LogHelper
+
+      before do
+        authenticated?
+      end
+
+      # ------------------------------------------------------------------------
+      # GET /helpdesk/stats/tickets
+      # ------------------------------------------------------------------------
+      desc 'Gets tickets stats for helpdesk within the duration specified'
+      params do
+        optional :from, type: DateTime, desc: 'The time to start getting statistics (do not provide for all stats)'
+        optional :to,   type: DateTime, desc: 'The time to stop getting statistics (defaults to current time)', default
+      end
+      get '/helpdesk/stats/tickets' do
+        unless authorise? current_user, HelpdeskTicket, :get_ticket_stats
+          error!({error: 'Not authorised to get helpdesk ticket stats'}, 403)
+        end
+
+        logger.info "#{current_user.username} got ticket helpdesk statistics"
+
+        from     = params[:from]
+        to       = params[:to] || DateTime.now
+
+        {
+          resolved_count:             HelpdeskTicket.resolved_between(from, to).length,
+          number_unresolved:          HelpdeskTicket.all_unresolved.length,
+          average_wait_time_in_mins:  HelpdeskTicket.average_wait_time(from, to)
+        }
+      end
+
+      # ------------------------------------------------------------------------
+      # GET /helpdesk/stats/dashgraph
+      # ------------------------------------------------------------------------
+      desc 'Gets dashboard graph data for the helpdesk (last 3 hours)'
+      get '/helpdesk/stats/dashgraph' do
+        unless authorise? current_user, HelpdeskTicket, :get_ticket_stats
+          error!({error: 'Not authorised to get helpdesk ticket stats'}, 403)
+        end
+
+        logger.info "#{current_user.username} got ticket helpdesk dash graph data"
+
+        # Work from now downto 3 hours ago
+        graph_range = 3.hours.ago.to_i..DateTime.now.to_i
+        graph_data = {}
+        # Collect data every minute going backwards -- stats for every min in
+        # last 180 mins
+        interval = 1.minute.to_i
+        graph_range.first.step(graph_range.last, interval) do |unix_t|
+          # Range is the time between graph_time less time mins ago
+          # to graph_time
+          range = Time.at(unix_t + interval).utc..Time.at(unix_t).utc
+          # Stats to include
+          avg_wait   = HelpdeskTicket.average_wait_time(range.first, range.last)
+          unresolved = HelpdeskTicket.unresolved_between(range.first, range.last)
+          # Insert at this time the stats there were
+          graph_data[unix_t] = {
+            average_wait_time_in_mins:    avg_wait,
+            number_of_unresolved_tickets: unresolved.length # count only
+          }
+        end
+        graph_data
+      end
+    end
+  end
+end
```

120 app/api/helpdesk/ticket.rb View

```
@@ -0,0 +1,120 @@
+require 'grape'
+require 'helpdesk_ticket_serializer'
+
+module Api
+  module Helpdesk
+    #
+    # Helpdesk ticketing system endpoints
+    #
+    class Ticket < Grape::API
+      helpers AuthHelpers
+      helpers AuthorisationHelpers
+      helpers LogHelper
+
+      before do
+        authenticated?
+      end
+
+     # ------------------------------------------------------------------------
+     # POST /helpdesk/tickets
+     # ------------------------------------------------------------------------
+     desc "Add a new helpdesk ticket"
+     params do
+       requires :project_id, type: Integer, desc: "The project to assign the ticket to"
+       optional :task_definition_id, type: Integer, desc: "Task which the student needs help with"
+       optional :description, type: String, desc: "Description associated to the ticket"
+     end
+     post '/helpdesk/tickets' do
+       project = Project.find(params[:project_id])
+       task = params[:task_definition_id].nil? ? nil : task = project.task_for_task_definition(params[:task_definition_
+
+       unless authorise? current_user, project, :create_ticket
+         error!({error: "Not authorised to create a ticket for project #{project.id}"}, 403)
+       end
+
+       # Only allow them to create a new ticket if they haven't done so already
+       if HelpdeskTicket.user_has_ticket_open? project.user.id
+         logger.info "#{current_user.username} tried to create new ticket but already has one open"
+         error!({error: "User already has a ticket open"}, 403)
+       end
+
+       ticket = HelpdeskTicket.create!({
+         project: project,
+         task: task,
+         description: params[:description]
+       })
+
+       logger.info "#{current_user.username} created new ticket #{ticket.id} for #{ticket.unit.name}"
+       ticket
+     end
+
+     # ------------------------------------------------------------------------
+     # GET /helpdesk/tickets?filter={resolved,unresolved,closed,all}&user_id={id}
+     # Optional User Id and Filter. Filter default to all.
+     # ------------------------------------------------------------------------
+     desc "Gets all helpdesk tickets. Optional User Id and Resolved filter."
+     params do
+       optional :user_id, type: String, desc: "The id of the user to get tickets for."
+       optional :filter, type: String, desc: "Filter by resolved, unresolved, closed or all. Defaults to all.", defaul
+       optional :shallow, type: Boolean, desc: "Use shallow serializer vs detailed serializer", default: true
+     end
+     get '/helpdesk/tickets' do
+       unless authorise? current_user, HelpdeskTicket, :get_tickets
+         error!({error: 'Not authorised to get tickets'}, 403)
+       end
+
+       user_id = params[:user_id]
+       filter = params[:filter] || 'all'
+       tickets = HelpdeskTicket.all_by_state(filter.to_sym, user_id)
+
+       serializer = params[:shallow] ? ShallowHelpdeskTicketSerializer : HelpdeskTicketSerializer
+       ActiveModel::ArraySerializer.new(tickets, each_serializer: serializer)
+     end
+
+     # ------------------------------------------------------------------------
+     # GET /helpdesk/tickets/:id
+     # ------------------------------------------------------------------------
+     desc "Gets helpdesk ticket with an id"
+     params do
+       requires :id, type: Integer, desc: "The id of the ticket to get"
+     end
+      get '/helpdesk/tickets/:id' do
+        ticket = HelpdeskTicket.find(params[:id])
+
+        if not authorise? current_user, ticket, :get_details
+          error!({error: 'Not authorised to get ticket details'}, 403)
+        end
+
+        ticket
+      end
+
+      # ------------------------------------------------------------------------
+      # DELETE /helpdesk/tickets/:id?[resolve=true|false]
+      # ------------------------------------------------------------------------
+      desc "Updates helpdesk ticket with an id"
+      params do
+        requires :id, type: Integer, desc: "The id to of the ticket to delete"
+        optional :resolve, type: Boolean, desc: "Mark the ticket as resolved"
+      end
+      delete '/helpdesk/tickets/:id' do
+        ticket = HelpdeskTicket.find(params[:id])
+
+        if params[:resolve] == true
+          unless authorise? current_user, ticket, :resolve_ticket
+            error!({error: "Not authorised to resolve ticket #{params[:id]}"}, 403)
+          end
+          ticket.resolve
+          logger.info "#{current_user.username} resolved ticket #{ticket.id}"
+        else
+          unless authorise? current_user, ticket, :close_ticket
+            error!({error: "Not authorised to close ticket #{params[:id]}"}, 403)
+          end
+          ticket.close
+          logger.info "#{current_user.username} closed ticket #{ticket.id}"
+        end
+
+        ticket
+      end
+    end
+  end
+end
```

179 app/models/helpdesk_session.rb View

```
@@ -0,0 +1,179 @@
+#
+# Tracking for sessions of staff working at the helpdesk.
+#
+class HelpdeskSession < ActiveRecord::Base
+  # Model associations
+  belongs_to :user
+
+  # Model constratins/validation
+  validates :user, presence: true
+  validates :clock_on_time, presence: true
+  validates :clock_off_time, presence: true
+
+  # Validates that a clock off time is always after the clock on time
+  validates_presence_of :clock_on_time, if: :clock_off_must_be_after_clock_on
+  def clock_off_must_be_after_clock_on
+    if clock_off_time && clock_off_time < clock_on_time
+      errors.add(:clock_off_time, "can't be before :clock_on_time")
+    end
+  end
```

```
+
+ # Validates that the user for a session is not a student
+ validates_presence_of :user, if: :user_is_not_student
+ def user_is_not_student
+   if user && user.role == Role.student
+     errors.add(:user, 'must not be a student')
+   end
+ end
+
+ #
+ # Permissions around helpdesk sessions
+ #
+ def self.permissions
+   # What can students do with sessions?
+   student_role_permissions = [
+     :get_all_current_session_users
+   ]
+   # What can tutors do with sessions?
+   tutor_role_permissions = [
+     :create_session,
+     :clock_off_session,
+     :get_all_current_session_users,
+     :get_sessions
+   ]
+   # What can convenors do with sessions?
+   convenor_role_permissions = [
+     :create_session,
+     :clock_off_session,
+     :get_all_current_session_users,
+     :get_sessions,
+     :get_ticket_stats
+   ]
+   # What can admins do with sessions?
+   admin_role_permissions = [
+     :create_session,
+     :clock_off_session,
+     :get_all_current_session_users,
+     :get_sessions,
+     :get_ticket_stats
+   ]
+   # What can nil users do with sessions?
+   nil_role_permissions = [
+   ]
+
+   # Return permissions hash
+   {
+     :admin    => admin_role_permissions,
+     :convenor => convenor_role_permissions,
+     :tutor    => tutor_role_permissions,
+     :student  => student_role_permissions,
+     :nil      => nil_role_permissions
+   }
+ end
+
+ def self.role_for(user)
+   user.role
+ end
+
+ def role_for(user)
+   # If the user provided is the user registered to the session, then
+   # they can do whatever they want with it. Convenors and admins can
+   # override this
+   if user == self.user || [Role.convenor, Role.admin].include?(user.role)
+     user.role
+   end
+ end
+
+ #
+ # Override the clock off time to set it to now
+ #
+ def clock_off
+   self.clock_off_time = DateTime.now
+   save!
+ end
+
+ #
+ # Returns the staff sessions within the given timeframe grouped by user id
+ #
+ def self.sessions_by_staff_between(from = nil, to = DateTime.now)
+   sessions_between(from, to).group_by(&:user)
+ end
+ def self.sessions_by_staff
+   sessions_by_staff_between
+ end
+
+ #
+ # Returns all session stats between the timeframe
+ #
+ def self.stats_by_staff_id(from = nil, to = DateTime.now)
+   sessions_by_staff_between(from, to).map do |user, sessions|
+     durations = sessions.map(&:session_duration)
+     average_duration = durations.inject(:+) / durations.length
+     count = sessions.length
+     {
+       user_id: user.id,
+       average_duration_in_hours: average_duration,
+       count: count
+     }
+   end
+ end
+
+ #
+ # Returns all sessions within the time-frame given
+ #
+ def self.sessions_between(from = nil, to = DateTime.now)
+   to ||= DateTime.now
+   return where('clock_on_time > ? AND clock_off_time < ?', from, to) unless from.nil?
+   where('clock_off_time < ?', to)
+ end
+
+ #
+ # Returns all currently active sessions
+ #
+ def self.active_sessions
+   where('clock_off_time > ?', DateTime.now)
+ end
+
+ #
+ # Returns all users currently working now
+ #
+ def self.users_working_now
+   active_sessions.map(&:user)
+ end
+
+ #
+ # Returns true if this session is clocked off
+  #
+  def clocked_off?
+    clock_off_time < DateTime.now
+  end
+
+  #
+  # Returns true if this session is clocked on
+  #
+  def clocked_on?
+    !clocked_off?
+  end
+
+  #
+  # Checks if the user provided is currently clocked off
+  #
+  def self.user_clocked_off?(user)
+    # Should have no sessions clocked on thus nothing found clocked on
+    user.helpdesk_sessions.find(&:clocked_on?).nil?
+  end
+
+  #
+  # Checks if the user provided is currently clocked on
+  #
+  def self.user_clocked_on?(user)
+    !user_clocked_off(user)
+  end
+
+  #
+  # Returns the duration of time for a session in hours
+  #
+  def session_duration
+    (clock_off_time - clock_on_time) / 60 / 60
+  end
+end
```

170 app/models/helpdesk_ticket.rb View

```
@@ -0,0 +1,170 @@
+#
+# Tracking for tickets logged at the helpdesk.
+#
+class HelpdeskTicket < ActiveRecord::Base
+  # Model associations
+  belongs_to :project
+  belongs_to :task
+
+  # Model constratins/validation
+  validates :project, presence: true # Must always be associated to a project
+
+  #
+  # Permissions around group data
+  #
+  def self.permissions
+    # What can students do with all tickets?
+    student_role_permissions = [
+      :get_details,
+      :get_tickets,
+      :get_ticket_stats,
+      :close_ticket
+    ]
+    # What can tutors|convenors|admins do with all tickets?
+    tutor_role_permissions = convenor_role_permissions = admin_role_permissions = [
+      :get_tickets,
+       :get_details,
+       :get_ticket_stats,
+       :close_ticket,
+       :resolve_ticket
+     ]
+     # What can nil users do with all tickets?
+     nil_role_permissions = [
+
+     ]
+
+     # Return permissions hash
+     {
+       :admin    => admin_role_permissions,
+       :convenor => convenor_role_permissions,
+       :tutor    => tutor_role_permissions,
+       :student  => student_role_permissions,
+       :nil      => nil_role_permissions
+     }
+   end
+
+   def self.role_for(user)
+     user.role
+   end
+
+   def role_for(user)
+     if user == project.user
+       Role.student
+     elsif user.role != Role.student
+       user.role
+     else
+       nil
+     end
+   end
+
+   # Returns true if a used has a ticket open
+   def self.user_has_ticket_open?(user_id)
+     !all_unresolved(user_id).empty?
+   end
+
+   # Returns back all unresolved tickets, optionally limit to a user
+   def self.all_unresolved(user_id = nil)
+     all_by_state(:unresolved, user_id)
+   end
+
+   # Returns back all resolved tickets, optionally limit to a user
+   def self.all_resolved(user_id = nil)
+     all_by_state(:resolved, user_id)
+   end
+
+   # Returns back all closed tickets, optionally limit to a user
+   def self.all_closed(user_id = nil)
+     all_by_state(:closed, user_id)
+   end
+
+   # Finds tickets of a particular status, optionally limit to a user
+   def self.all_by_state(resolved_filter, user_id = nil)
+     tickets =
+       case resolved_filter
+       when :resolved
+         where(is_resolved: true)
+       when :unresolved
+         where(is_closed: false)
+       when :closed
+        where(is_closed: true, is_resolved: false)
+      else
+        all
+      end
+
+    unless user_id.nil?
+      user = User.find(user_id)
+      project = Project.for_user(user, false) # see app/models/project.rb:76
+      tickets = tickets.where(project: project) # limits the scope of tickets down to those with the project provided
+    end
+
+    tickets.order(:created_at)
+  end
+
+  # Get all tickets resolved between two dates
+  def self.resolved_between(from = nil, to = DateTime.now)
+    to ||= DateTime.now # if nil is passed in
+    from ? all_resolved.where(closed_at: from..to) : all_resolved
+  end
+
+  #
+  # Get all tickets that were still unresolved between from and to
+  # dates
+  #
+  def self.unresolved_between(from, to = DateTime.now)
+    to ||= DateTime.now # if nil passed in
+    where('created_at <= ? AND (closed_at >= ? OR NOT is_closed)', from, to)
+  end
+
+  # Calculates the average time to resolve a ticket from the duration
+  # provided (i.e., between from and now). If no arguments are provided,
+  # all resolved tickets will be used regardless of when they were resolved.
+  # Where no tickets are found within the period, nil is returned.
+  def self.average_resolve_time_between(from = nil, to = DateTime.now)
+    tickets = resolved_between(from, to)
+    resolved_between(from, to).average(:minutes_to_resolve).to_i unless tickets.empty?
+  end
+
+  #
+  # Determines the current average wait time for a ticket
+  #
+  def self.average_wait_time(from, to)
+    avg_resolve = HelpdeskTicket.average_resolve_time_between(from, to) || 0
+    num_tickets = HelpdeskTicket.unresolved_between(from, to).length
+    avg_resolve * num_tickets
+  end
+
+  # Resolves the ticket
+  def resolve
+    unless is_closed
+      # Resolving closes the ticket
+      close
+      self.is_resolved = true
+      self.minutes_to_resolve = ((closed_at - created_at) / 60).to_f.round(2)
+      save!
+    end
+  end
+
+  # Unit for ticket
+  def unit
+    project.unit
+  end
+
+  # Student for ticket
+  def student
+    project.student
+  end
+
+  # Returns true if ticket is associated with a task
+  def task?
+    !task.nil?
+  end
+
+  # Prematurely closes a ticket
+  def close
+    unless is_closed
+      self.is_closed = true
+      self.closed_at = DateTime.now
+      save!
+    end
+  end
+end
```

7 app/models/project.rb

View

```
@@ -21,6 +21,8 @@ class Project < ActiveRecord::Base
 
   has_many :learning_outcome_task_links, through: :tasks
 
+  has_many :helpdesk_tickets, dependent: :destroy
+
   validate :must_be_in_group_tutorials
 
   #
@@ -33,7 +35,8 @@ def self.permissions
       :change_tutorial,
       :make_submission,
       :get_submission,
-      :change
+      :change,
+      :create_ticket
     ]
     # What can tutors do with projects?
     tutor_role_permissions = [
@@ -860,6 +863,8 @@ def has_task_for_task_definition?(td)
   # task if the task does not exist for this project.
   #
   def task_for_task_definition(td)
+    # Find the definition if a ID is passed in.
+    td = TaskDefinition.find(td) if td.is_a? Fixnum
     logger.debug "Finding task #{td.abbreviation} for project #{log_details()}"
     result = tasks.where(task_definition: td).first
     if result.nil?
```

1 app/models/task.rb

View

```
@@ -74,6 +74,7 @@ def role_for(user)
   has_many :reverse_plagiarism_match_links, class_name: "PlagiarismMatchLink", dependent: :destroy, inverse_of: :other_
   has_many :learning_outcome_task_links, dependent: :destroy # links to learning outcomes
   has_many :learning_outcomes,  through: :learning_outcome_task_links
+  has_many :helpdesk_tickets
 
   validates :task_definition_id, uniqueness: { scope: :project,
     message: "must be unique within the project" }
```

5 app/models/user.rb View

```
@@ -67,6 +67,7 @@ def reset_authentication_token!
   belongs_to  :role   # Foreign Key
   has_many    :unit_roles, dependent: :destroy
   has_many    :projects
+  has_many    :helpdesk_sessions

   # Model validations/constraints
   validates :first_name,  presence: true
@@ -407,4 +408,8 @@ def self.import_from_csv(current_user, file)
       errors:  errors
     }
   end
+
+  def units_taught
+    unit_roles.map(&:unit)
+  end
 end
```

15 app/serializers/helpdesk_session_serializer.rb View

```
@@ -0,0 +1,15 @@
+require 'user_serializer'
+
+class HelpdeskSessionSerializer < ActiveModel::Serializer
+  attributes :id,
+             :user,
+             :clocked_on?,
+             :clock_on_time,
+             :clock_off_time
+
+  has_one :user, serializer: HelpdeskUserSerializer
+
+  def clocked_on?
+    object.clocked_on?
+  end
+end
```

40 app/serializers/helpdesk_ticket_serializer.rb View

```
@@ -0,0 +1,40 @@
+require 'project_serializer'
+require 'task_definition_serializer'
+
+class HelpdeskTicketSerializer < ActiveModel::Serializer
+  attributes :id,
+             :project,
+             :task_definition,
+             :description,
+             :is_resolved,
+             :is_closed,
+             :created_at,
+             :closed_at,
+             :minutes_to_resolve
+
+  def project
+    ShallowProjectSerializer.new object.project
+  end
+
+  def task_definition
+    task = object.task
+    ShallowTaskDefinitionSerializer.new task.task_definition if task
+  end
+end
+
+class ShallowHelpdeskTicketSerializer < ActiveModel::Serializer
+  attributes :id,
+             :project_id,
+             :task_definition_id,
+             :description,
+             :is_resolved,
+             :is_closed,
+             :created_at,
+             :closed_at,
+             :minutes_to_resolve
+
+  def task_definition_id
+    task = object.task
+    task.task_definition_id if task
+  end
+end
```

25 app/serializers/project_serializer.rb View

```
@@ -10,6 +10,31 @@ class ShallowProjectSerializer < ActiveModel::Serializer
   def project_id
     object.id
   end
+
+  def student_name
+    object.student.name
+  end
+
+  def unit_id
+    object.unit.id
+  end
+
+  def unit_code
+    object.unit.code
+  end
+
+  def unit_name
+    object.unit.name
+  end
+
+  def tutor_name
+    tutorial = object.tutorial
+    tutorial.tutor.name if tutorial
+  end
+
+  def start_date
+    object.unit.start_date.to_date
+  end
 end

 # Student project serializer is used with teaching staff
```

4 app/serializers/task_definition_serializer.rb View

```
@@ -1,3 +1,7 @@
+class ShallowTaskDefinitionSerializer < ActiveModel::Serializer
+  attributes :id, :abbreviation, :name, :description
```

```
+end
+
 class TaskDefinitionSerializer < ActiveModel::Serializer
   attributes :id, :abbreviation, :name, :description,
     :weight, :target_grade, :target_date,
```

8 app/serializers/user_serializer.rb View

```
@@ -10,3 +10,11 @@ def system_role
 class ShallowUserSerializer < ActiveModel::Serializer
   attributes :id, :name, :email
 end
+
+class HelpdeskUserSerializer < ActiveModel::Serializer
+  attributes :id, :name, :email, :units_taught
+
+  def units_taught
+    ActiveModel::ArraySerializer.new(object.units_taught, each_serializer: ShallowUnitSerializer)
+  end
+end
```

15 db/migrate/20160705025331_create_helpdesk_tickets.rb View

```
@@ -0,0 +1,15 @@
+class CreateHelpdeskTickets < ActiveRecord::Migration
+  def change
+    create_table :helpdesk_tickets do |t|
+      t.references :project, null: false
+      t.references :task, null: true
+
+      t.string :comments, null: true
+      t.boolean :is_resolved, null: false, default: false
+
+      t.timestamps
+    end
+    add_index :helpdesk_tickets, :project_id
+    add_index :helpdesk_tickets, :task_id
+  end
+end
```

6 db/migrate/20160708011422_change_helpdesk_ticket_comment_to_description.rb View

```
@@ -0,0 +1,6 @@
+class ChangeHelpdeskTicketCommentToDescription < ActiveRecord::Migration
+  def change
+    rename_column :helpdesk_tickets, :comments, :description
+    change_column :helpdesk_tickets, :description, :string, :limit => 2048
+  end
+end
```

11 db/migrate/20160722114921_create_helpdesk_sessions.rb View

```
@@ -0,0 +1,11 @@
+class CreateHelpdeskSessions < ActiveRecord::Migration
+  def change
+    create_table :helpdesk_sessions do |t|
+      t.references :user, null: false
+      t.datetime :clock_on_time, null: false
+      t.datetime :clock_off_time, null: false
+      t.timestamps
+    end
+    add_index :helpdesk_sessions, :user_id
+  end
+end
```

5 db/migrate/20160725141011_add_resolved_at_to_helpdesk_ticket.rb View

```
@@ -0,0 +1,5 @@
+class AddResolvedAtToHelpdeskTicket < ActiveRecord::Migration
+  def change
+    add_column :helpdesk_tickets, :resolved_at, :datetime
+  end
+end
```

5 db/migrate/20160725144833_add_minutes_to_resolve_to_helpdesk_ticket.rb View

```
@@ -0,0 +1,5 @@
+class AddMinutesToResolveToHelpdeskTicket < ActiveRecord::Migration
+  def change
+    add_column :helpdesk_tickets, :minutes_to_resolve, :float
+  end
+end
```

5 db/migrate/20160826233551_add_is_closed_to_helpdesk_ticket.rb View

```
@@ -0,0 +1,5 @@
+class AddIsClosedToHelpdeskTicket < ActiveRecord::Migration
+  def change
+    add_column :helpdesk_tickets, :is_closed, :boolean, :default => false
+  end
+end
```

5 db/migrate/20160910065932_rename_resolved_at_to_closed_at.rb View

```
@@ -0,0 +1,5 @@
+class RenameResolvedAtToClosedAt < ActiveRecord::Migration
+  def change
+    rename_column :helpdesk_tickets, :resolved_at, :closed_at
+  end
+end
```

27 db/schema.rb View

```
@@ -11,7 +11,7 @@
 #
 # It's strongly recommended that you check this file into your version control system.

-ActiveRecord::Schema.define(version: 20160503222451) do
+ActiveRecord::Schema.define(version: 20160910065932) do

   # These are extensions that must be enabled in order to support this database
   enable_extension "plpgsql"
@@ -74,6 +74,31 @@

   add_index "helpdesk_schedules", ["user_id"], name: "index_helpdesk_schedules_on_user_id", using: :btree

+  create_table "helpdesk_sessions", force: :cascade do |t|
+    t.integer  "user_id",        null: false
+    t.datetime "clock_on_time",  null: false
+    t.datetime "clock_off_time", null: false
+    t.datetime "created_at"
+    t.datetime "updated_at"
+  end
+
+  add_index "helpdesk_sessions", ["user_id"], name: "index_helpdesk_sessions_on_user_id", using: :btree
```

```
+
+  create_table "helpdesk_tickets", force: :cascade do |t|
+    t.integer  "project_id",                                      null: false
+    t.integer  "task_id"
+    t.string   "description",        limit: 2048
+    t.boolean  "is_resolved",                     default: false, null: false
+    t.datetime "created_at"
+    t.datetime "updated_at"
+    t.datetime "closed_at"
+    t.float    "minutes_to_resolve"
+    t.boolean  "is_closed",                       default: false
+  end
+
+  add_index "helpdesk_tickets", ["project_id"], name: "index_helpdesk_tickets_on_project_id", using: :btree
+  add_index "helpdesk_tickets", ["task_id"], name: "index_helpdesk_tickets_on_task_id", using: :btree
+
   create_table "learning_outcome_task_links", force: :cascade do |t|
     t.text     "description"
     t.integer  "rating"
```

131 lib/helpers/database_populator.rb View

```
@@ -2,6 +2,7 @@
 require 'faker'
 require 'bcrypt'
 require 'json'
+require_all 'lib/helpers'

 #
 # This class populates data in the database
@@ -29,7 +30,7 @@ def initialize(scale = :small)
         some_tutorials: 1,
         many_tutorials: 1,
         max_tutorials: 4,
-        tickets_generated: 10
+        tickets_to_generate: 30
       },
       large: {
         min_students: 15,
@@ -41,7 +42,7 @@ def initialize(scale = :small)
         some_tutorials: 2,
         many_tutorials: 4,
         max_tutorials: 20,
-        tickets_generated: 50
+        tickets_to_generate: 100
       }
     }
     accepted_scale_types = scale_data.keys
@@ -135,14 +136,80 @@ def generate_units
       generate_and_align_ilos_for_unit(unit, unit_details)
       generate_tutorials_and_enrol_students_for_unit(unit, unit_details)
     end
+    puts "!"
   end

   #
-  # Random project helper
+  # Generates some units
   #
-  def random_project
-    id = Project.pluck(:id).sample
-     Project.find(id)
+   def generate_units
+     puts "--> Generating units"
+
+     if @user_cache.empty?
+       # Must generate users first!
+       puts "---> No users generated. Generating users first..."
+       generate_users()
+     end
+
+     # Set sizes from scale
+     some_tasks = @scale[:some_tasks]
+     many_tasks = @scale[:many_tasks]
+     some_tutorials = @scale[:some_tutorials]
+     many_tutorials = @scale[:many_tutorials]
+
+     # Run through the unit_details and initialise their data
+     @unit_data.each do | unit_key, unit_details |
+       puts "---> Generating unit #{unit_details[:code]}"
+       unit = Unit.create!(
+         code: unit_details[:code],
+         name: unit_details[:name],
+         description: Populator.words(10..15),
+         start_date: Time.zone.now  - 6.weeks,
+         end_date: 13.weeks.since(Time.zone.now - 6.weeks)
+       )
+       # Assign the convenors for this unit
+       unit_details[:convenors].each do | user_key |
+         puts "----> Adding convenor #{user_key}"
+         unit.employ_staff(@user_cache[user_key], Role.convenor)
+       end
+       # Cache what we have
+       @unit_cache[unit_key] = unit
+       # Generate other unit-related stuff
+       generate_tasks_for_unit(unit, unit_details)
+       generate_and_align_ilos_for_unit(unit, unit_details)
+       generate_tutorials_and_enrol_students_for_unit(unit, unit_details)
+     end
+   end
+
+   #
+   # Generates helpdesk tickets
+   #
+   def generate_helpdesk_tickets
+     tickets_to_generate = @scale[:tickets_to_generate]
+     print "-> Generating #{tickets_to_generate} helpdesk tickets"
+     tickets_to_generate.times do
+       project = Randomizer.random_record_for_model(Project)
+       throw "Must create projects before calling generate_helpdesk_tickets" if project.nil?
+       # 1/4 chance of getting nil task
+       task = rand(0..3) == 0 ? nil : Randomizer.random_task_for_project(project)
+       # 3/4 chance of being resolved
+       is_resolved = rand(0..3) > 0
+       if is_resolved
+         minutes_to_resolve = rand(0..15)
+         closed_at = DateTime.now + minutes_to_resolve.minutes
+       end
+       HelpdeskTicket.create(
+         project: project,
+         task: task,
+         description: Populator.words(5..10),
+         is_resolved: is_resolved,
+         is_closed: is_resolved,
+       closed_at: closed_at,
+       minutes_to_resolve: minutes_to_resolve
+     )
+     print "."
+   end
+   puts "!"
  end

  private
@@ -323,42 +390,22 @@ def generate_tutorials_and_enrol_students_for_unit(unit, unit_details)
  end

  #
- # Finds or creates a student if not found
- #
- def find_or_create_student(username)
-   unless @user_cache.has_key?(username)
-     profile = {
-       first_name:             Faker::Name.first_name,
-       last_name:              Faker::Name.last_name,
-       nickname:               username,
-       role_id:                Role.student_id,
-       email:                  "#{username}@doubtfire.com",
-       username:               username,
-       password:               'password',
-       password_confirmation:  'password'
-     }
-     user = User.create!(profile)
-     @user_cache[username] = user
-   end
-   @user_cache[username]
- end
-
- #
  # Generated fixed data here for students and units
  #
  def generate_fixed_data
    # Define fixed user data here
    @user_data = {
      acain:              {first_name: "Andrew",  last_name: "Cain",          nickname: "Macite",         role_id: Role
-     cwoodward:          {first_name: "Clinton", last_name: "Woodward",      nickname: "The Giant",      role_id: Role
-     ajones:             {first_name: "Allan",   last_name: "Jones",         nickname: "P-Jiddy",        role_id: Role
+     aconvenor:          {first_name: "Clinton", last_name: "Woodward",      nickname: "The Giant",      role_id: Role
+     aadmin:             {first_name: "Allan",   last_name: "Jones",         nickname: "P-Jiddy",        role_id: Role
      rwilson:            {first_name: "Reuben",  last_name: "Wilson",        nickname: "Reubs",          role_id: Role
-     akihironoguchi:     {first_name: "Akihiro", last_name: "Noguchi",       nickname: "Animations",     role_id: Role
+     atutor:             {first_name: "Akihiro", last_name: "Noguchi",       nickname: "Animations",     role_id: Role
      acummaudo:          {first_name: "Alex",    last_name: "Cummaudo",      nickname: "DoubtfireDude",  role_id: Role
      cliff:              {first_name: "Cliff",   last_name: "Warren",        nickname: "Cliff",          role_id: Role
      joostfunkekupper:   {first_name: "Joost",   last_name: "Funke Kupper",  nickname: "Joe",            role_id: Role
      angusmorton:        {first_name: "Angus",   last_name: "Morton",        nickname: "Angus",          role_id: Role
-     "123456X" =>        {first_name: "Fred",    last_name: "Jones",         nickname: "Foo",            role_id: Role
+     "123456X" =>        {first_name: "Fred",    last_name: "Jones",         nickname: "Foo",            role_id: Role
+     astudent:           {first_name: "student", last_name: "surname",       nickname: "Foo",            role_id: Role
    }
    # Add 10 tutors to fixed info
    10.times do |count|
@@ -381,14 +428,14 @@ def generate_fixed_data
      intro_prog: {
        code: "COS10001",
        name: "Introduction to Programming",
-       convenors: [ :acain, :cwoodward ],
+       convenors: [ :acain, :aconvenor ],
          tutors: [
            { user: :acain, num: many_tutorials },
-           { user: :cwoodward, num: many_tutorials },
-           { user: :ajones, num: many_tutorials },
+           { user: :aconvenor, num: many_tutorials },
+           { user: :aadmin, num: many_tutorials },
            { user: :rwilson, num: many_tutorials },
            { user: :acummaudo, num: some_tutorials },
-           { user: :akihironoguchi, num: many_tutorials },
+           { user: :atutor, num: many_tutorials },
            { user: :joostfunkekupper, num: many_tutorials },
            { user: :angusmorton, num: some_tutorials },
            { user: :cliff, num: some_tutorials },
@@ -400,11 +447,11 @@ def generate_fixed_data
        oop: {
          code: "COS20007",
          name: "Object Oriented Programming",
-         convenors: [ :acain, :cwoodward, :ajones, :acummaudo ],
+         convenors: [ :acain, :aconvenor, :aadmin, :acummaudo ],
          tutors: [
            { user: "tutor_1", num: few_tutorials },
            { user: :angusmorton, num: few_tutorials },
-           { user: :akihironoguchi, num: few_tutorials },
+           { user: :atutor, num: few_tutorials },
            { user: :joostfunkekupper, num: few_tutorials },
          ],
          num_tasks: many_tasks,
@@ -414,9 +461,9 @@ def generate_fixed_data
        ai4g: {
          code: "COS30046",
          name: "Artificial Intelligence for Games",
-         convenors: [ :cwoodward ],
+         convenors: [ :aconvenor ],
          tutors: [
-           { user: :cwoodward, num: few_tutorials },
+           { user: :aconvenor, num: few_tutorials },
            { user: :cliff, num: few_tutorials },
          ],
          num_tasks: few_tasks,
@@ -426,13 +473,13 @@ def generate_fixed_data
        gameprog: {
          code: "COS30243",
          name: "Game Programming",
-         convenors: [ :cwoodward, :acummaudo ],
+         convenors: [ :aconvenor, :acummaudo ],
          tutors: [
-           { user: :cwoodward, num: few_tutorials },
+           { user: :aconvenor, num: few_tutorials },
          ],
          num_tasks: few_tasks,
          ilos: rand(0..3),
-         students: [ :acain, :ajones ]
+         students: [ :acain, :aadmin ]
        },
      }
    puts "-> Defined #{@user_data.length} fixed users and #{@unit_data.length} units"
```

22 lib/helpers/find_or_create_students.rb View

```
@@ -0,0 +1,22 @@
+#
```

```
+# Finds or creates a student if not found
+#
+def find_or_create_student(username)
+  user_created = nil
+  using_cache = !!@user_cache
+  if not using_cache or not @user_cache.has_key?(username)
+    profile = {
+      first_name:             Faker::Name.first_name,
+      last_name:              Faker::Name.last_name,
+      nickname:               username,
+      role_id:                Role.student_id,
+      email:                  "#{username}@doubtfire.com",
+      username:               username,
+      password:               'password',
+      password_confirmation:  'password'
+    }
+    user_created = User.create!(profile)
+    @user_cache[username] = user_created if using_cache
+  end
+  user_created || @user_cache[username]
+end
```

25 lib/helpers/randomizer.rb View

```
@@ -0,0 +1,25 @@
+class Randomizer
+  #
+  # Randomly returns a new model (e.g., random_record_for_model(Project))
+  #
+  def self.random_record_for_model(model)
+    throw "Must provide a model" unless model.is_a? Class
+    id = model.all.pluck(:id).sample
+    model.find(id)
+  end
+
+  #
+  # Randomly returns a new task for the given project
+  #
+  def self.random_task_for_project(project)
+    task_def = self.random_task_def_for_project(project)
+    project.task_for_task_definition(task_def)
+  end
+
+  #
+  # Randomly returns a new task definition for the given project
+  #
+  def self.random_task_def_for_project(project)
+    project.unit.task_definitions.sample
+  end
+end
```

26 lib/tasks/populate.rake View

```
@@ -2,32 +2,11 @@ require_all 'lib/helpers'
 namespace :db do
   desc "Mark off some of the due tasks"
   task expand_first_unit: :environment do
-
-    find_or_create_student = lambda { |username|
-      result = User.find_by_username(username)
-      if result
-        return result
-      else
-         profile = {
-           first_name:             Faker::Name.first_name,
-           last_name:              Faker::Name.last_name,
-           nickname:               username,
-           role_id:                Role.student_id,
-           email:                  "#{username}@doubtfire.com",
-           username:               username,
-           password:               'password',
-           password_confirmation:  'password'
-         }
-
-         result = User.create!(profile)
-         return result
-       end
-     }
-
      unit = Unit.first
      tutes = unit.tutorials
      for student_count in 0..2000
-       proj = unit.enrol_student(find_or_create_student.call("student_#{student_count}"), tutes[student_count % tutes.co
+       student = find_or_create_student("student_#{student_count}")
+       proj = unit.enrol_student(student, tutes[student_count % tutes.count])
      end
    end

@@ -177,6 +156,7 @@ namespace :db do
      dbpop = DatabasePopulator.new ENV['SCALE']
      dbpop.generate_users()
      dbpop.generate_units()
+     dbpop.generate_helpdesk_tickets()

      # Run simulate signoff?
      unless !args.nil? && args[:extend_populate]
```

2 lib/tasks/test_setup.rake View

```
@@ -6,8 +6,8 @@ namespace :test do
    task setup: [:environment, 'db:setup', 'db:migrate'] do
      require 'helpers/database_populator'
      dbpop = DatabasePopulator.new
-     dbpop.generate_user_roles()
      dbpop.generate_users()
      dbpop.generate_units()
+     dbpop.generate_helpdesk_tickets()
    end
  end
```

17 test/api/auth_test.rb View

```
@@ -1,6 +1,6 @@
 require 'test_helper'

-class AuthTest < MiniTest::Test
+class AuthTest < ActiveSupport::TestCase
   include Rack::Test::Methods
   include TestHelpers::AuthHelper
   include TestHelpers::JsonHelper
@@ -38,10 +38,7 @@ def test_auth_post
     # These match the model object... so can compare in loops
     user_keys = [ 'id', 'email', 'first_name', 'last_name', 'username', 'nickname', 'receive_task_notifications', 'rece

```

```diff
-     assert_json_matches_model(response_user_data, expected_auth, user_keys)
-
-     user_keys.each { |k| assert response_user_data.has_key?(k), "Response has key #{k}" }
-     user_keys.each { |k| assert_equal expected_auth[k], response_user_data[k], "Values for key #{k} match" }
+     assert_json_matches_model(response_user_data, expected_auth, *user_keys)

      # Check other values returned
      assert_equal expected_auth.name, response_user_data['name'], 'Names match'
@@ -72,28 +69,28 @@ def test_auth_roles
        {
          expect: Role.admin,
          post: {
-             username: "acain",
+             username: "aadmin",
              password: "password"
          }
        },
        {
          expect: Role.convenor,
          post: {
-             username: "jrenzella",
+             username: "aconvenor",
              password: "password"
          }
        },
        {
          expect: Role.tutor,
          post: {
-             username: "rwilson",
+             username: "atutor",
              password: "password"
          }
        },
        {
          expect: Role.student,
          post: {
-             username: "acummaudo",
+             username: "astudent",
              password: "password"
          }
        }
@@ -116,7 +113,6 @@ def test_auth_roles

    # Test put for authentication token
    def test_auth_put
-     auth_token = get_auth_token
      data_to_put = {
          username: "acain",
          password: "password"
@@ -136,7 +132,6 @@ def test_auth_put

    # Test for deleting authentication token
    def test_auth_delete
-     auth_token = get_auth_token
      # Get the auth token needed for delete test
      delete "/api/auth/#{auth_token}.json", "CONTENT_TYPE" => 'application/json'
      # 200 response code means success!
```

95 test/api/helpdesk/ticket_test.rb View

```diff
@@ -0,0 +1,95 @@
```

```
+require 'test_helper'
+
+class TicketsTest < ActiveSupport::TestCase
+  include Rack::Test::Methods
+  include TestHelpers::AuthHelper
+  include TestHelpers::JsonHelper
+
+  def app
+    Rails.application
+  end
+
+  # POST /helpdesk/tickets
+  def test_post_helpdesk_ticket
+    project = Randomizer.random_record_for_model(Project)
+
+    # Can't create ticket if this user already has a ticket
+    while HelpdeskTicket.user_has_ticket_open? project.user_id
+      project = Randomizer.random_record_for_model(Project)
+    end
+
+    ticket = {
+      project_id: project.id,
+      description: Populator.words(5..10),
+      task_definition_id: rand > 0.33 ? Randomizer.random_task_def_for_project(project).id : nil
+    }
+
+    data_to_post = add_auth_token ticket, project.user
+
+    post_json '/api/helpdesk/tickets', data_to_post
+
+    # Last ticket ids should match
+    expected_ticket = HelpdeskTicket.last
+    assert_json_matches_model last_response_body, expected_ticket, :id
+  end
+
+  # GET /helpdesk/tickets
+  def test_get_all_helpdesk_tickets
+    get with_auth_token '/api/helpdesk/tickets'
+    expect = ActiveModel::ArraySerializer.new(HelpdeskTicket.all, each_serializer: ShallowHelpdeskTicketSerializer)
+    assert_json_equal expect, last_response_body
+  end
+
+  # GET /helpdesk/tickets
+  def test_get_all_unresolved_helpdesk_tickets
+    get with_auth_token '/api/helpdesk/tickets?filter=unresolved'
+    expect = ActiveModel::ArraySerializer.new(HelpdeskTicket.all_unresolved, each_serializer: ShallowHelpdeskTicketSeri
+    assert_json_equal expect, last_response_body
+  end
+
+  # GET /helpdesk/tickets?filter=resolved
+  def test_get_resolved_helpdesk_tickets
+    get with_auth_token '/api/helpdesk/tickets?filter=resolved'
+    expect = ActiveModel::ArraySerializer.new(HelpdeskTicket.all_resolved, each_serializer: ShallowHelpdeskTicketSerial
+    assert_json_equal expect, last_response_body
+  end
+
+  # GET /helpdesk/tickets?user_id=1&filter=resolved
+  def test_get_resolved_helpdesk_tickets_for_user
+    id = 1
+    get with_auth_token "/api/helpdesk/tickets?user_id=#{id}&filter=resolved"
+    expect = ActiveModel::ArraySerializer.new(HelpdeskTicket.all_resolved(id), each_serializer: ShallowHelpdeskTicketSe
+    assert_json_equal expect, last_response_body
+  end
```

```
+
+  # GET /helpdesk/tickets/:id
+  # id = 1
+  def test_get_helpdesk_ticket_with_id
+    id = 1
+    get with_auth_token "/api/helpdesk/tickets/#{id}"
+    expect = HelpdeskTicketSerializer.new HelpdeskTicket.find(id)
+    assert_json_equal expect, last_response_body
+  end
+
+  # DELETE /helpdesk/tickets/:id?resolve=false
+  def test_delete_a_ticket_by_closing
+    ticket = HelpdeskTicket.all_unresolved.first
+    delete with_auth_token "/api/helpdesk/tickets/#{ticket.id}"
+    # Reload the ticket from DB with updated info
+    ticket = HelpdeskTicket.find(ticket.id)
+    assert ticket.is_closed
+    refute ticket.is_resolved
+  end
+
+  # DELETE /helpdesk/tickets/:id?resolve=true
+  def test_delete_a_ticket_by_resolving
+    ticket = HelpdeskTicket.all_unresolved.first
+    delete with_auth_token "/api/helpdesk/tickets/#{ticket.id}?resolve=true"
+    # Reload the ticket from DB with updated info
+    ticket = HelpdeskTicket.find(ticket.id)
+    assert ticket.is_resolved
+    assert ticket.is_closed
+    dt = DateTime.parse(last_response_body['closed_at']).inspect
+    assert dt, ticket.closed_at
+  end
+end
```

12 test/api/units_test.rb

View

```
@@ -1,7 +1,7 @@
 require 'test_helper'
 require 'date'

-class UnitsTest < MiniTest::Test
+class UnitsTest < ActiveSupport::TestCase
   include Rack::Test::Methods
   include TestHelpers::AuthHelper
   include TestHelpers::JsonHelper
@@ -10,10 +10,6 @@ def app
     Rails.application
   end

-  def setup
-    @auth_token = get_auth_token()
-  end
-
   # --------------------------------------------------------------------------- #
   # --- Endpoint testing for:
   # ------- /api/units.json
@@ -100,7 +96,7 @@ def test_units_post_tutorial
   # Test GET for getting all units
   def test_units_get
     # The GET we are testing
-    get "/api/units.json?auth_token=#{@auth_token}"
+    get with_auth_token "/api/units.json"

     actual_unit = JSON.parse(last_response.body)[0]
```

```diff
     expected_unit = Unit.first
@@ -124,7 +120,7 @@ def test_units_get_by_id
     # Get response back from getting a unit by id
 
     # Test getting the first unit with id of 1
-    get  "/api/units/1.json?auth_token=#{@auth_token}"
+    get with_auth_token "/api/units/1.json"
 
     actual_unit = JSON.parse(last_response.body)
     expected_unit = Unit.find(1)
@@ -137,7 +133,7 @@ def test_units_get_by_id
 
     # Get response back from getting a unit by id
     # Test getting the first unit with id of 2
-    get  "/api/units/2.json?auth_token=#{@auth_token}"
+    get with_auth_token "/api/units/2.json"
 
     actual_unit = JSON.parse(last_response.body)
     expected_unit = Unit.find(2)
```

33 test/helpers/auth_helper.rb View

```diff
@@ -4,25 +4,40 @@ module TestHelpers
   #
   module AuthHelper
     #
-    # Gets an authentication token for User.first
+    # Gets an auth token for the provided user
     #
-    def get_auth_token
-      user = User.first
+    def auth_token(user = User.first)
       user.extend_authentication_token(true)
       user.auth_token
     end
 
     #
-    # Adds an authentication token to the hash of data provided
+    # Adds an authentication token to the hash data or string URL
     # This prevents us from having to keep adding the :auth_token
-    # key to any POST data that is needed
+    # key to any GET/POST/PUT etc. data that is needed
     #
-    def add_auth_token(hash)
-      hash[:auth_token] = get_auth_token
-      hash
+    def add_auth_token(data, user = User.first)
+      # Passed in an id instead of a user model? Find the user model from User.find
+      user = User.find(user) if user.is_a? Fixnum
+      if data.is_a? Hash
+        data[:auth_token] = auth_token user
+      elsif data.is_a? String
+        # If we have a question mark, we need to add a query paramater using &
+        # otherwise use ?
+        data << (data.include?('?') ? "&" : "?") << "auth_token=#{auth_token}"
+      end
+      data
     end
 
-    module_function :get_auth_token
+    #
+    # Alias for above for nicer usage (e.g., get with_auth_token "http://")
+    #
+    def with_auth_token(data, user = User.first)
+      add_auth_token data, user
+    end
+
+    module_function :auth_token
     module_function :add_auth_token
+    module_function :with_auth_token
   end
 end
```

33 test/helpers/json_helper.rb View

```diff
@@ -20,15 +20,40 @@ def put_json(endpoint, data)
     end
 
     #
-    # Assert that a JSON response matches the model and keys provided
+    # Assert that a JSON response matches the model and keys provided.
+    # Keys MUST be symbols
     #
-    def assert_json_matches_model(response_json, model, keys)
-      keys.each { |k| assert response_json.has_key?(k), "Response has key #{k}"}
-      keys.each { |k| assert_equal model[k], response_json[k], "Values for key #{k} match" }
+    def assert_json_matches_model(response_json, model, *keys)
+      keys.each { |k| assert response_json.key?(k.to_s), "Response has key #{k}"}
+      keys.each { |k| assert_equal model[k], response_json[k.to_s], "Values for key #{k} match" }
+    end
+
+    #
+    # Last response body parsed from JSON
+    #
+    def last_response_body
+      JSON.parse(last_response.body)
+    end
+
+    #
+    # Converts from an ActiveRelation to JSON (without Ruby objects inside the hash)
+    #
+    def json_hashed(hash)
+      JSON.parse(hash.to_json)
+    end
+
+    #
+    # Assert that the lefthand matches the right-hand as json hash
+    #
+    def assert_json_equal(lhs, rhs)
+      assert_equal json_hashed(lhs), json_hashed(rhs)
     end
 
     module_function :assert_json_matches_model
     module_function :post_json
     module_function :put_json
+    module_function :last_response_body
+    module_function :json_hashed
+    module_function :assert_json_equal
   end
 end
```

68 test/models/helpdesk/ticket_test.rb View

```diff
...   ...   @@ -0,0 +1,68 @@
```

```
+require "test_helper"
+
+class HelpdeskTicketTest < ActiveSupport::TestCase
+  setup do
+    @test_desc = 'Hello, World!'
+    @test_project = Randomizer.random_record_for_model(Project)
+    @test_task = Randomizer.random_task_for_project(@test_project)
+    @tickets = {
+      without_task: HelpdeskTicket.create!(
+        project: @test_project
+      ),
+      with_task: HelpdeskTicket.create!(
+        project: @test_project,
+        task: @test_task
+      ),
+      with_description: HelpdeskTicket.create!(
+        project: @test_project,
+        description: @test_desc
+      )
+    }
+  end
+
+  test "helpdesk tickets have projects" do
+    @tickets.values.each do |t|
+      assert t.project, @test_project
+    end
+  end
+
+  test "only one helpdesk ticket has a description" do
+    assert_not_nil @tickets[:with_description].description
+    assert_equal @tickets[:with_description].description, @test_desc
+    assert_nil @tickets[:with_task].description
+    assert_nil @tickets[:without_task].description
+  end
+
+  test "only one helpdesk ticket has a task" do
+    assert_equal @tickets[:with_task].task, @test_task
+    assert_nil @tickets[:with_description].task
+    assert_nil @tickets[:without_task].task
+  end
+
+  test "resolving a helpdesk ticket" do
+    ticket_to_resolve = @tickets[:without_task]
+    ticket_to_resolve.resolve
+    assert ticket_to_resolve, true
+    # both without_task and ticket_to_resolve should match
+    assert_equal @tickets[:without_task], ticket_to_resolve
+    # others should remain false
+    refute @tickets[:with_task].is_resolved
+    refute @tickets[:with_description].is_resolved
+  end
+
+  test "the task? method should be true if the task has a task" do
+    assert @tickets[:with_task].task?
+  end
+
+  test "the student method should match the associated project's student" do
+    assert_equal @tickets[:with_task].student, @test_project.student
+    assert_equal @tickets[:with_task].student, @test_project.student
+    assert_equal @tickets[:with_description].student, @test_project.student
+  end
+
+  test "the unit method should match the associated project's unit" do
+    assert_equal @tickets[:with_task].unit, @test_project.unit
+    assert_equal @tickets[:with_task].unit, @test_project.unit
+    assert_equal @tickets[:with_description].unit, @test_project.unit
+  end
+end
```

18 test/test_helper.rb View

```
@@ -17,16 +17,14 @@
 # Require minitest extensions
 require 'minitest/rails'
 require 'minitest/pride'
-require 'minitest/autorun'
 require 'minitest/osx'
+require 'minitest/around'

-# Require all test helpers
-require_all 'test/helpers'
+# Require test helpers
 require 'rails/test_help'
 require 'database_cleaner'

 class ActiveSupport::TestCase
-  # Check if migrations are pending
   ActiveRecord::Migration.check_pending!

   # Setup all fixtures in test/fixtures/*.(yml|csv) for all tests in alphabetical order.
@@ -41,17 +39,17 @@ class ActiveSupport::TestCase
   # Support rollback of db changes after all tests
   DatabaseCleaner.strategy = :transaction

-  # After setup of all test, start database cleaner to undo transactions
-  def before_teardown
-    super
+  def setup
     DatabaseCleaner.start
   end

-  # After teardown
-  def after_teardown
+  def teardown
     DatabaseCleaner.clean
-    super
   end

   # Add more helper methods to be used by all tests here...
+  require_all 'test/helpers'
+
+  extend MiniTest::Spec::DSL
+
 end
```